UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

BRENO TRINDADE TOSTES
LEONARDO VENTURA CABRAL LEAL

# Healthy aging: A data-driven approach to Indicators of Compromise decaying models

RIO DE JANEIRO
2021

BRENO TRINDADE TOSTES
LEONARDO VENTURA CABRAL LEAL

# Healthy aging: A data-driven approach to Indicators of Compromise decaying models

Undergraduate final paper presented to the Computing Institute at the Federal University of Rio de Janeiro as a prerequisite to receive the bachelor's degree in Computer Science.

Supervisor: Prof. Dr. Daniel Sadoc Menasché
Co-supervisor: Dr. Enrico Lovat

RIO DE JANEIRO
2021

Tostes, Breno Trindade

T716h

Healthy aging: a data-driven approach to Indicators of Compromise decaying models / Breno Trindade Tostes, Leonardo Ventura Cabral Leal. – Rio de Janeiro, 2021.

46 f.

Orientador: Daniel Sadoc Menasché.
Coorientador: Enrico Lovat.

Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) - Universidade Federal do Rio de Janeiro, Instituto de Computação, Bacharel em Ciência da Computação, 2021.

1. Segurança da informação. 2. Cibersegurança. 3. Indicadores de comprometimento. 4. Modelos de TTL. I. Leal, Leonardo Ventura Cabral. II. Menasché, Daniel Sadoc (Orient.). III. Lovat, Enrico (Coorient.). IV. Universidade Federal do Rio de Janeiro, Instituto de Matemática. V. Título.

BRENO TRINDADE TOSTES
LEONARDO VENTURA CABRAL LEAL

Healthy aging: A data-driven approach to Indicators of Compromise decaying models

Trabalho de conclusão de curso de graduação apresentado ao Instituto de Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 22 de novembro de 2021

BANCA EXAMINADORA:

Daniel Sadoc Menasché

Daniel Sadoc Menasché, Ph.D.
(UMass)

Cláudio Miceli de Farias, D.Sc.
(UFRJ)

gov.br PAULO HENRIQUE DE AGUIAR RODRIGUES
Data: 01/12/2021 08:23:37-0300
Verifique em https://verificador.iti.br

Paulo Henrique de Aguiar Rodrigues, Ph.D.
(UCLA)

Documento assinado digitalmente
gov.br SILVANA ROSSETTO
Data: 28/11/2021 11:40:47-0300
Verifique em https://verificador.iti.br

Silvana Rossetto, D.Sc.
(PUC-RJ)

## ACKNOWLEDGEMENTS

First, we would like to thank UFRJ for giving us the education and opportunities to pursue this research.

Our profound respect and thanks to Prof. Dr. Daniel Menasché and Dr. Enrico Lovat, who guided us throughout this work providing immeasurable insight and cultivating our growth during such a crucial period of our lives.

I dedicate this work to those who have offered me with all the support I needed and to those who might reap the fruits of my labor one day. My deepest gratitude to my family, who stood with me during the warm moments which inspire us with love, but also during the harshest times, when uncertainties creep in and, as if on a whim, threaten to claim all that we take for granted.

**Breno Tostes**

I would like to thank my best friend and partner, Mylena, who supported me during my entire journey at UFRJ. I appreciate every advice you gave me during this period of intense dedication.

To Maridir, Cacá, Luciane, Evaldo, Fabiane and all my family and friends for the support you provided me during all these years and for shaping me into who I am today.

To Luna, Mary, Leoa, Chico, Minnie, Magali, Fred and Simba for all the love you gave me.

To Alex, for inspiring me into pursuing a computer science career.

To Daniel, for all the support, advice and friendship during the majority of the time I spent at UFRJ.

To Matheus, Leandro and Enrico for the friendship and professional support to ensure I could thrive.

To Miguel, Mateus, Abhishek and everyone on our research group for the discussions and insights provided on so many different venues.

To all my friends at GRIS, SIGA and UFRJ for being brilliant colleagues.

To everyone who was not directly mentioned, but helped me get here.

**Leonardo Ventura**

*"I do see the beauty in the rules, the invisible code of chaos hiding behind the menacing face of order."*

**Elliot Alderson**

# RESUMO

Indicadores de Comprometimento (IoC) são a base do campo de inteligência de ameaças. Eles são utilizados em monitoradores de rede, gerando alertas quando uma correspondência é encontrada, permitindo que seja possível reagir a essas ameaças. No entanto, uma quantidade enorme de IoCs são gerados todo dia, tornando impossível monitorar cada IoC na mesma escala a longo prazo, além de aumentar a possibilidade de gerar alertas falsos. Neste trabalho, nos utilizamos de dados de rede reais de IoCs, e seus avistamentos, para modelar o decaimento da pontuação de IoCs ao longo do tempo. Começamos com a caracterização do nosso conjunto de dados e explicamos suas especificidades. Em seguida, apresentamos nossos modelos de tempo de vida (TTL), que podem receber como parâmetro uma porcentagem aceitável de perdas de avistamentos ou custos associados de monitoramento e de perda de um avistamento. Quando os valores absolutos dos custos associados ao monitoramento e perdas não estão disponíveis, mas a razão entre os mesmos pode ser estimada, propomos um terceiro modelo a ser adotado. Dada a razão entre custos, e o traço de avistamentos, o modelo fornece limiares além dos quais medidas extremas passam a ser ótimas. Em particular, quando a razão entre custos é menor que o limiar inferior, sempre monitorar todos os IoCs passa a ser ótimo. Similarmente, quando a razão entre custos é maior que o limiar superior calculado usando o modelo, a estratégia ótima consiste em nunca monitorar os IoCs.

**Palavras-chave**: Segurança da Informação; Cibersegurança; Indicadores de Comprometimento; Modelos de TTL;

# ABSTRACT

Indicators of Compromise (IoCs) are the foundation of cyber threat intelligence. They are employed for monitoring purposes, generating alerts when a match is found and allowing experts to act accordingly. However, an unbearable number of IoCs are generated everyday, making it impossible to monitor every IoC in the same scale in the long term, and also increasing the possibility of generating false alerts. In this work, we leverage a real world trace of IoCs and its sightings to model the decrease of IoC scoring over time. We start with a characterization of our dataset and explain its specifics. Then, we present time-to-live (TTL) models, which take as input a percentage of acceptable misses or associated costs of monitoring and missing. When monitoring and missing costs are not available, but their ratio can still be estimated, the model can also provide thresholds where, if the ratio between cost of monitoring and missing is below or above them, always or never monitoring is the best solution.

**Keywords**: Cybersecurity; Cyber Threat Intelligence; Indicators of Compromise; TTL models;

## LIST OF FIGURES

# LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| CTI | Cyber Threat Intelligence |
| TI | Threat Intelligence |
| OSINT | Open Source Intelligence |
| IoC | Indicator of Compromise |
| IR | Incident Response |
| MISP | Malware Information Sharing Platform |
| SIEM | Security Information and Event Management |
| IDS | Intrusion Detection System |
| SOC | Security Operations Center |
| TTL | Time to live |
| CDF | Cumulative Distribution Function |
| ASTI | Azure Sentinel Threat Intelligence |

# LIST OF SYMBOLS

| | |
|---|---|
| $T$ | Initial TTL |

## SUMMARY

# 1 INTRODUCTION

Cyber Threat Intelligence (CTI) is a field of Cybersecurity (or Information Security, both are used here interchangeably). Its main goal is to gather intelligence about current threats that could compromise a system. In the current scenario, CTI plays a major role in cybersecurity. By using CTI, companies aim to protect themselves against known threats. However, CTI acts as a rather reactive approach by leveraging available information and therefore does not attempt to defend against unknown attacks, such as zero-days[1].

Threat intelligence comes from a variety of sources, both public and private data feeds are used to aggregate and share information between communities. Publicly available information is usually gathered using Open Source Intelligence (OSINT), while private data is fruit of agreements between companies, threat intelligence platforms, governments and even non-governmental organizations.

OSINT is currently known as a framework for aggregating intelligence from public sources, such as companies' websites, public data feeds, technical articles(LIAO et al., 2016), and even social media (SABOTTKE; SUCIU; DUMITRAș, 2015; MITTAL et al., 2016; LE et al., 2019).

Indicators of compromise (IoCs) are the foundation of CTI. In practice, they are used to gather information about possible attacks being conducted and as a way of detecting which machines have possibly been infected by these artifacts. IoCs can be of different types, such as hashes, ip addresses, domain names, etc, and they can also contain associated tags, a field following standardized taxonomies, used to represent additional information about an IoC and its context, e.g. tlp:white to indicate the IoC can be shared with the general public.

Another application of IoCs lies on Incident Response (IR), a field responsible for handling incidents. In IR, IoCs are used to check if a machine has been compromised with a known attack vector, e.g. checking if the hash of a suspicious program matches any hash identified as an IoC through Threat Intelligence.

In order to gather and maintain IoCs, platforms such as the Malware Information Sharing Platform (MISP)(WAGNER et al., 2016) are implemented. These platforms are employed to share IoC – not only malware – information between entities. Also, on MISP, it is possible to add sightings or false positive indicators to IoCs. Sightings, in general, indicate that a specific IoC has been seen in the network. They are created by automated systems or manually inserted by Security Operations Center (SOC) analysts. Alternatively, an analyst can also indicate that an IoC is in fact a false positive, meaning that alerts should no longer be generated if a match is found. As a result, a sighting can provide

[1] Zero-days are attacks where the vendor is not aware of the vulnerability being exploited in its product and there is no patch available to mitigate its risks.

insight into the freshness and validity of an IoC. To detect possible attacks being conducted, companies usually use a Security Information and Event Management (SIEM) and Intrusion Detection Systems (IDS) to automatically generate alerts and identify sightings of IoCs based on logs.

Threat Intelligence feeds (TI feeds) is a term to describe a curated source of IoCs. There are public and private feeds, where a feed itself is defined as a list of events shared with feed's users. An event generally represents a campaign conducted by a threat actor and each event contains a list of associated IoCs. An IoC can be in multiple events. An example of event is a phishing[2] campaign. Corresponding IoCs could be an email subject, email source, hash of a malicious file attachment, etc.

## 1.1 THE PROBLEM

Monitoring too few IoCs may threaten the security of the target environment by missing sightings in related events. On the other hand, maintaining too many IoCs is prohibitive due to the intrinsic cost of investigating a large catalog of potential incidents (LIAO et al., 2016). Table 1 summarizes the considered scenario.

Table 1 – Problem description

| | monitoring many IoCs | monitoring few IoCs |
|---|---|---|
| many missed sightings | unlikely | high risks |
| few missed sightings | potentially false alarms and alarm fatigue | desired outcome |

The costs related to monitoring too many IoCs involve the increased number of false positives, intrinsic limitations of SOC employees, as well as monetary costs related to the price model. Azure Sentinel Threat Intelligence, for instance, offers two alternatives: Capacity Reservations and Pay-As-You-Go. In the latter, the current cost is of $2.46 per GB-ingested. If the presence of IoCs is used to pre-filter the data to be fed to Azure, the larger the number of IoCs being monitored, the larger the incurred costs (MICROSOFT, 2021b; MICROSOFT, 2021a).

Various solutions have been proposed in the community to model the lifecycle of IoCs and to define aging models to automatically determine *expiration dates* for indicators. However, to the best of our knowledge, the evaluations of these models is limited and speculative in nature. In part, this occurs due to lack of data to support the models.

In this work, we report results of a study that aimed at answering our key motivating questions:

- How long should a certain indicator be monitored for?

[2] Malicious attack that tries to induce users to act against their own safety, e.g. click a malicious file attachment in an email.

- What's the optimal aging model for a given environment?

To tackle those questions we leveraged sightings collected from a real environment, for more than one year, and used them to compare different aging models. We refer to the two main parameters of the model as the missing and monitoring costs, related to the impact of missing a sighting and to the attention span required to handle alarms, respectively.

Among our trace-driven findings, we discover that if the ratio between monitoring over missed costs is between two thresholds, namely 1/476,197,152 and 1/312 if we consider only sightings, the system benefits from storing IoCs for a finite time-to-live, which can be set according to the IoC category. To the best of our knowledge, this is the first real world evaluation of thresholds related to IoC aging. Then, we identify and discuss assumptions behind the considered model. Although the precise numbers are tailored for the considered environment, we envision that the fundamental insights provided in this work generalize to designers tackling the trade-offs involved in the assessment of IoC monitoring and aging.

## 1.2 DYNAMICS OF IOC CREATION AND SIGHTINGS

Figure 1 shows the typical dynamics of IoC creation and sightings. An IoC is typically discovered at a vendor or TI source, e.g., in a controlled environment or through a honeypot. Then, the IoC is created and published by the vendor at its platform. At this point, the information is propagated to (MISP) instances. The creation date of the IoC corresponds to the date at which it was created at the corresponding MISP instance. Finally, SIEM systems monitor for matches of IoCs in the network, and eventually report sightings.

Remark: as will be further discussed in Section 3.2.3 some IoCs may have sightings before their corresponding creation dates.

Figure 2 illustrates the dynamics of IoC creations and sightings on a given timeline. Arrows above the black line represent timestamps where either an IoC was created (creation date, dashed arrow) or sighting (regular arrow) happened. The time between the creation date and the first sighting is represented by a dashed block, while a regular block represents time between sightings of the same IoC.

## 1.3 CONTRIBUTIONS

In summary, our key contributions are the following:

1. **Measurement insights:** by leveraging real traces of IoC sightings, we report statistics of interest, e.g., on the time between IoC creation and first sighting, and on how those statistics vary across IoC categories

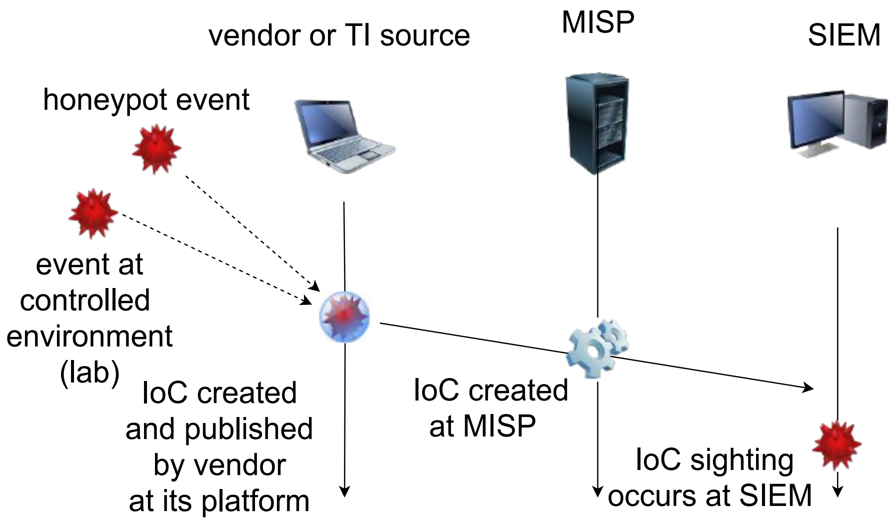
vendor or TI source
MISP
SIEM
honeypot event
event at controlled environment (lab)
IoC created and published by vendor at its platform
IoC created at MISP
IoC sighting occurs at SIEM

Figure 1 – Representation of typical dynamics of IoC creation and sightings.

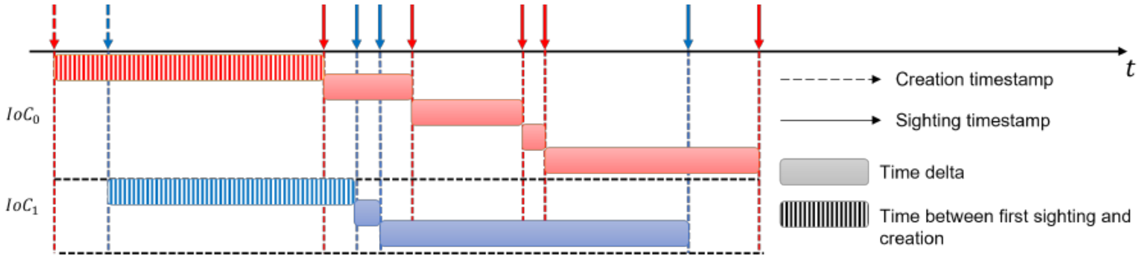

Figure 2 – A simplified visualization of a categorically divided trace.
Red lane: symbolizes a `md5` IoC and its timestamps.
Blue lane: symbolizes a `sha1` type IoC and its timestamps.

2. **Model based assessment of aging:** we evaluate time-to-live (TTL) aging models for IoCs, indicating how the collected statistics can be used to parametrize those model and shed insights on the thresholds to decide when and if an IoC should be evicted. The qualitative discussion of the assumptions behind the aging models can assist practitioners in determining how to maintain their pool of IoCs, whereas the quantitative analysis provides the first known ballpark references regarding temporal aspects associated to the interplay between IoC creation, sightings and maintenance.

## 1.4 OUTLINE

In the following chapter, we analyze related work and their overlap with ours. Then, on Chapter 3, we describe the dataset, its limitations, singularities, and discuss discovered insights. On Chapter 4, we present and discuss the TTL models proposed. Finally, we conclude on Chapter 5 with final considerations and future work.

# 2 RELATED WORK

Next, we compare our work with previous research done in the field. First, we analyze the IoC landscape and the usage of information sharing platforms. Then, we analyze previous approaches to IoC aging and its implementation details in comparison to ours. Last, we analyze techniques related to TTL modeling. A summary of how our work relates to prior art is presented in Table 2.

Table 2 – Related work

| | contains sightings | no sightings |
|---|---|---|
| contextual information<br>(textual description) | unaware<br>(future work) | NLP (LIAO et al., 2016)<br>(BOUWMAN et al., 2020) |
| all context anonymized<br>(no textual descriptions) | this work | - |

## 2.1 CHARACTERIZATION OF TI FEEDS AND IOCS

Previous work on TI feeds and IoC usage generally focused either on gathering intelligence from sources to provide more insight into possible threats or on understanding the current usage of TI feeds.

In (LIAO et al., 2016), authors present iACE, an automated solution to extract IoCs from text. Their approach leverages the predictable structure of IoCs in technical articles, usually indicated by context terms. Once a match on this structure is found, an IoC in the OpenIOC format – a machine readable format for IoC usage – is generated, describing not only its artifact, but also its context. Their approach had a coverage of over 90% and precision of 95% on articles written in English. More recently, (LONG et al., 2019) proposed a neural based model to automatically identify IoCs from articles with the usage of a multi-head attention module and contextual features. Such model can gather contextual information from cybersecurity articles and can identify IoCs with an average F1-score of 89.0% on English articles an 81.5% on Chinese's.

An automated technique to find and validate IoCs for web applications using a data driven approach is presented in (CATAKOGLU; BALDUZZI; BALZAROTTI, 2016). By analyzing information collected from a honeypot, authors were able to develop a way to detect malicious websites and generate corresponding IoCs, such as the IP address of the site. Generally, not all compromised pages are malicious, e.g. a website showing a message from a hacker is not inherently malicious. However, detecting those and actually malicious websites is paramount to enforce safety on the web. Also, the use of IoCs for malicious websites, like drive-by-download pages, allows detection and correlation of such pages before more traditional approaches. Catakogly et al showed on their experiment that their system was able to generate IoCs for websites infected for months with no previously known detection.

Additionally, (MITTAL et al., 2016; DIONÍSIO et al., 2019) use Twitter as a source of CTI due to its intrinsic real time behavior and strong information security community. While in (MITTAL et al., 2016; SABOTTKE; SUCIU; DUMITRAȘ, 2015) authors present a way of using Twitter as a general OSINT source – not only IoC information, in (DIONÍSIO et al., 2019; ALVES et al., 2021) authors introduce a way of using cybersecurity content on Twitter to produce a security alert or fill an IoC.

In (SABOTTKE; SUCIU; DUMITRAȘ, 2015), authors conduct a quantitative and qualitative analysis of vulnerability information on Twitter and describe a technique for early detection of real world exploitation using social media. They also introduce a threat model and test its robustness against three different adversarial interference. Moreover, in (MITTAL et al., 2016), Semantic Web RDF is used to represent intelligence produced and SWRL rules to assess relevancy of the extracted information and issuance of alerts to analysts. In summary, they developed a framework to serve users with alerts where they used Security Vulnerability Concept Extractor (SVCE) to retrieve terms related to vulnerabilities. Then, they store such information in a cybersecurity knowledge database as an RDF. Then, they use SWRL rules to create generate alerts based on a user profile with infrastructural details.

Alternatively, in (DIONÍSIO et al., 2019), a processing pipeline of cybersecurity-related tweets using deep neural networks is shown. First, a convolutional neural network identify relevant cybersecurity tweets. After, a bidirectional long short-term memory (LSTM) network extracts named entities from these previously identified tweets, finally culminating in the filling of an IoC or a security alert. Across three case study infrastructures, the described pipeline achieves an average of 94% for true positive and 91% for false negative rates for the classification task and an average F1-score of 92% for the named entity recognition part. In (ALVES et al., 2021), SYNAPSE, a streaming threat monitor able to continuously generate a summary of the threat landscape relevant to a monitored infrastructure, is introduced. SYNAPSE's pipeline comprises filtering, feature extraction, binary classification and clustering to generate IoCs. With an experiment ran for an 8 month period with more than 195,000 tweets from 80 accounts, they were able to achieve a 90% rate for true positive cases and, based on CVSS(MELL; SCARFONE; ROMANOSKY, 2006) and the availability of patches or exploits, generated IoCs were relevant and timeliness to analysts.

Furthermore, (BOUWMAN et al., 2020) show an empirical assessment of commercial threat intelligence feeds. By comparing two leading vendors, authors indicate that these services have almost no overlap neither between them, nor among them and other four other large public TI feeds. Even limiting to 22 specific threat actors both vendors claim to track, only 2.5% to 4% of indicators overlap on average, which shows a huge heterogeneity among these TI feeds. In addition, by conducting 14 interviews with security professionals who use these services, authors found that customers of paid TI feeds value the better curated and more selective paid TI sources over other sources due to a general noise attached to public TI feeds. For them, a curated source consumes less analyst time and a potentially limited coverage is of lesser concern.

More recently, (BOUWMAN et al., 2022) gathered a security information sharing volunteer community with over 4,000 members to form the COVID-19 Cyber Threat Coalition. They addressed recurrent questions on threat information sharing, such as: *does collaboration at scale lead to better coverage?* and *does making threat information freely available improve defenders' ability to react?*. Although its focus shifted from COVID-19 (1.4% - 3.6%) to more generic threats, like phishing, in the partition of data related to COVID, they found evidence that such communities do generate impact: abuse detection infrastructures were aware of only 25.1% domains listed on CTC, compared to 58.4% on the overall dataset (not only related to COVID).

However, these approaches are still geared towards *generating* IoCs, i.e. they could be used to generate the dataset we use, while our goal is to trim unnecessary IoCs off of the system. Next, we analyze related work with such goal.

## 2.2 IOC DECAYING MODELS

Using decaying models to improve IoC's storage is not generally the default approach. As a result, most previous work on the subject is still primitive and there is not much literature available.

In (IKLODY et al., 2018) the authors set forth a number of assumptions which are not clearly derived from data, such as: 1) customers have 1 week time to fix web-servers; and 2) typical blacklists take 48 hours to be applied in proxy servers or browsers. In this work, in contrast, we aim at deriving our results directly from data.

Furthermore, (IKLODY et al., 2018) show a generic model for decreasing IoC score over time and its implementation on MISP. Their model is calculated based on the following variables, specific to each IoC:

- *base_score*: weighting of tags and confidence in the source
- *elapsed_time*: elapsed time between first and last sighting
- $\tau$: time when the overall score should be zero
- *decay_rate*: score decreasing speed

Further, *score* is defined as:

$$score = base\_score - decay\_rate(T_t - T_{t-1}) \tag{2.1}$$

which means that $score$ starts at $base\_score$ and decreases over time in accordance to $decay\_rate$. When $score = 0$, the IoC is no longer relevant and can be evicted from the system. Also, $base\_score \in [0, 100]$ and whenever a new sighting occurs, the resulting $score$ is reset to $base\_score$.

Additionally, $base\_score$ is defined as:

$$base\_score = weight * tags + w_{sc} * source\_confidence \tag{2.2}$$

where weight of each $tag \in [0, 1]$ is defined in its predicate level, and $w_{sc} \in [0, 1]$ is used to consider subtle trust evaluations, such as organization's current reputability. It is important to note that such reputability could be subject to change if an organization has been recently compromised, for example. Those two metrics are complementary so $weight + w_{sc} = 100$

The $decay\_rate$ is subject to a decaying function over time and depends on the type of the IoC (e.g. IP address, hash, filename, etc). If the IoC is an IP address, for example, its decaying function should be slower in the first few hours and get steeper after some time to reflect the regular use case where after an IP is shared among the community, more users can start blocking it, consequently reducing the attack's effectiveness and the importance of the associated IoC. Also, since IP are an ephemeral intelligence, it could be reassigned to a legitimate service and generate innocuous alerts, taking analysts' precious time.

Authors analyzed multiple scenarios and their correspondent parameters for $base\_score$ and $decay_rate$, generating different scores for each proposed scenario. However, these parameters used are based on assumptions made by their extensive experience, and not on data.

In (ERMERINS et al., 2020), authors focused on improving scoring models by taking multiple feeds into account. They also used decay function and source confidence parameters to reach a final score.

To calculate scores across multiple feeds, they used AbuseIPDB(AbuseIPDB, 2021), Binary Defense Banlist(Binary Defense, 2021), C&C Tracker(CONSULTING, 2021) and Cyber Cure free intelligence feed(CYBERCURE, 2021). Since all of these TI feeds provide IP addresses, they limited their scope to a final scoring function specific for IP addresses. Although such feeds do show independence, which is necessary to ensure the final score is not biased towards a certain feed, they have a reduced overlap – the largest overlap, between AbuseIPDB and CyberCure, is 7.5% – limiting the number of IoCs available for analysis.

Additionally, authors propose a new $source\_confidence$, result of a combination of multiple feeds and a final score as follows:

$$final\_score = \frac{\sum_{i=0}^{N} source\_confidence_i{}^2 * score_i}{\sum_{i=0}^{N} source\_confidence_i} \tag{2.3}$$

## 2.3 TTL MODEL

TTL models have been widely used for caching systems (CARRA; NEGLIA; MICHIARDI, 2019; MOURA et al., 2019). Here, we highlight related work in terms of TTL model usage, despite none of them having a previously established relationship with IoCs and their heterogeneous nature. Nevertheless, the TTL model's principle applies to our use case: some data is deemed valid until a TTL elapses and it turns irrelevant.

Fagin (FAGIN, 1977) was the first to show that TTL models can be used to mimic the behavior of FIFO and LRU caches. In (JUNG; BERGER; BALAKRISHNAN, 2003), authors developed a closed-form formula for the hit ratio for a sequence of events where the time between events is independent and identically distributed. They provide a cache-hit rate analysis of TTL-based caches in terms of statistics of data accesses and the defined TTL. By using DNS traces, they find the proposed model is good at predicting observed statistics, e.g. it explains why the hit ratio for a 15 minute TTL is over 80% and increasing it from 15 minutes to 24 hours only increases the hit ratio by less than 17%. Additionally, they find that if the inter-arrival times for DNS requests are modeled by an analytic distribution, a Pareto distribution with a point mass performs better than any other candidate distribution.

# 3 TRACE CHARACTERIZATION

In this section we describe and characterize our trace of IoCs and sightings generated in a real world environment. Although the results discussed in this chapter are specific to our trace, the methods utilized for the characterization are generalizable. The IoCs and sightings considered are generated based on the dynamics discussed on Section 1.2.

## 3.1 DATASET STRUCTURE

Our trace has, in total, over 14,000,000 IoCs being monitored with no aging factors, i.e., no IoC is evicted from monitoring at any point in time. However, only 5,789 of them have at least one sighting and 2,635 have more than one sighting. In our calculations, we take the entire dataset into account, but this analysis will focus solely on those with at least one associated sighting.

Each row in our dataset represents one sighting. For each sighting, we have:

- `sighting-date`: the timestamp in which the sighting occured
- `ioc-id`: id of the IoC that generated the sighting (anonymized)
- `creation-date`: date when the IoC was created in MISP's system
- `category`: the IoC's category
- `event-id`: the associated event (anonymized)
- `tags`: a list of tags, each of which provides additional details about the IoC (anonymized)

A sample of the dataset is shown in table 3. Due to a non-disclosure agreement, however, the trace is composed of anonymized data, making the use of highly descriptive information such as tags less viable.

## 3.2 TEMPORAL ANALYSIS

Although the first entry for an IoC with at least one sighting dates to 09/09/2018, until 24/04/2019 no sightings were registered (see Figure 3), as the feature allowing sightings to be registered in MISP's system was only implemented in April of 2019. During the period between 24/04/2019 and 02/09/2020, 892,240 sightings were observed with hourly granularity, i.e., the time of a sighting can only be traced back to a specific hour on a specific date in which it occurs.

In Figure 3 we show the CDF of sightings and creation dates. In the first few months we spot a huge increase in sightings being created. Interestingly, 66% of all sightings

Table 3 – Dataset visualization example

| **ioc-id** | **event-id** | **category** | **creation-date** | **sighting-date** | **tags** |
|---|---|---|---|---|---|
| 0 | 8794 | domain | 2019-09-30 15:26:00 | 2019-11-05 21:00:00 | [Tag-5, Tag-6, ..., Tag-1079] |
| 1 | 7976 | sha256 | 2019-07-26 16:30:00 | 2019-07-27 03:00:00 | [Tag-17, Tag-13, ... , Tag-16] |
| 2 | 7976 | filename | 2019-07-26 16:30:00 | 2019-07-27 02:00:00 | [Tag-17, Tag-13, ... , Tag-16] |
| ... | ... | ... | ... | ... | ... |
| 5788 | 17362 | url | 2020-08-24 16:30:00 | 2020-09-02 21:00:00 | [Tag-5, Tag-13, ... , Tag1079] |
| 5788 | 17362 | url | 2020-08-24 16:30:00 | 2020-09-02 21:00:00 | [Tag-5, Tag-13, ... , Tag1079] |

happened around the first 3 months, as shown by the red line. After, we see a shift in behavior and the CDF shows a slower rate of sightings, with a few spikes indicating a burst in their insertion. Such burst could be explained by a late creation of sightings accumulated during the flat phase preceding such spikes.

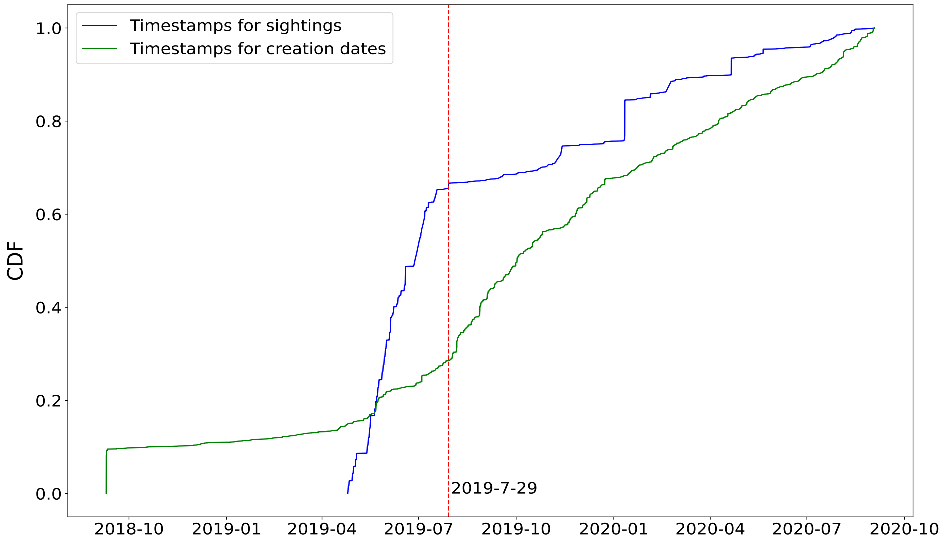

Figure 3 – CDF of creation dates & sightings over observed period

### 3.2.1 IoC creations and sightings occur roughly uniformly over days of the week and months

In Figure 4 we show the number of sightings and creation dates by day of the week. As we can see, in green, there is usually fewer sightings on Sunday compared to other days of the week. Considering creation dates of each IoC, in blue, we see a mostly equivalent behavior, except a huge decrease on Saturdays, which could be attributed to some system maintenance. More interesting, however, is that most sightings occurred on IoCs where the creation date was a Sunday, as seen by the orangebar.

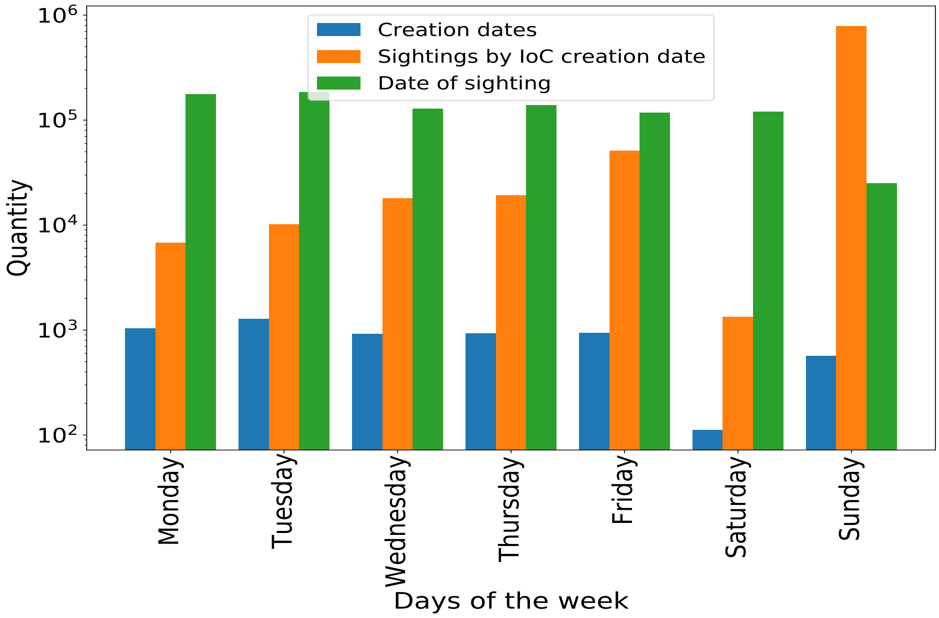

Figure 4 – Distribution of sightings and creation date by day of the week

Considering months, Figure 5 shows a higher sighting activity mostly in May and June. August and September were the months where most IoCs were created. Also, by more than an order of magnitude, IoCs created in September are the ones with most associated sightings, due to the nature of indicators created in the month of September that produced abnormal amounts of sightings, skewing the temporal analysis.

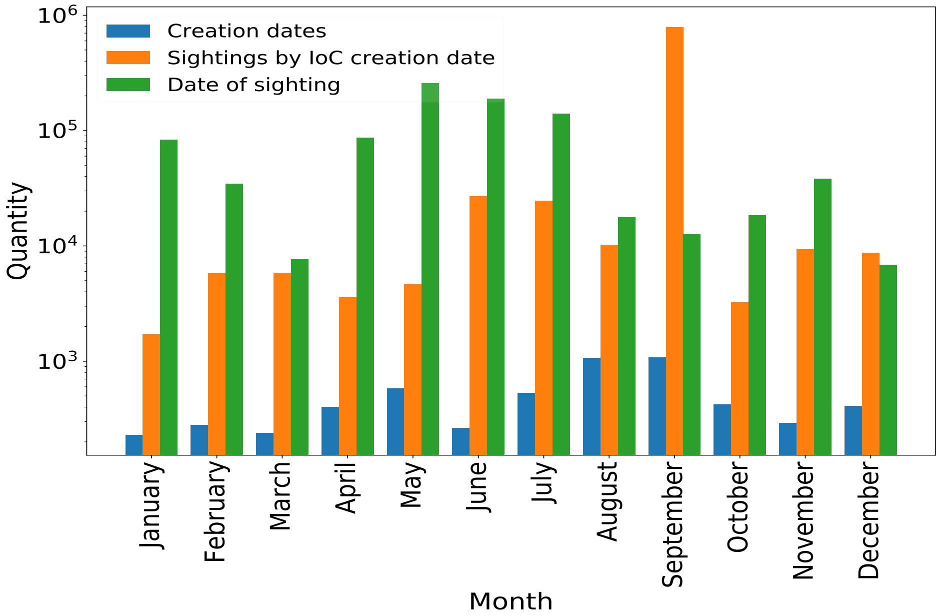

Figure 5 – Distribution of sightings and creation date by month

### 3.2.2 A small fraction of IoCs is responsible for a large number of sightings

The spikes of IoCs created in August and September are then explained by Figures 6 and 7. 785,436 of all sightings (88.02%) are registered by IoCs created in 09/09/2018, these previous plots show the skew created by this concentration of sightings, which is explained by a bulk insertion of IoCs into the system. In Figure 6, we see IoCs with most associated sightings. 14 out of the top 25 and 9 out of the top 10 were created in 09/09/2018. On the other end, out of the 5,789 IoCs with at least one sighting, 3,154 have exactly one sighting.

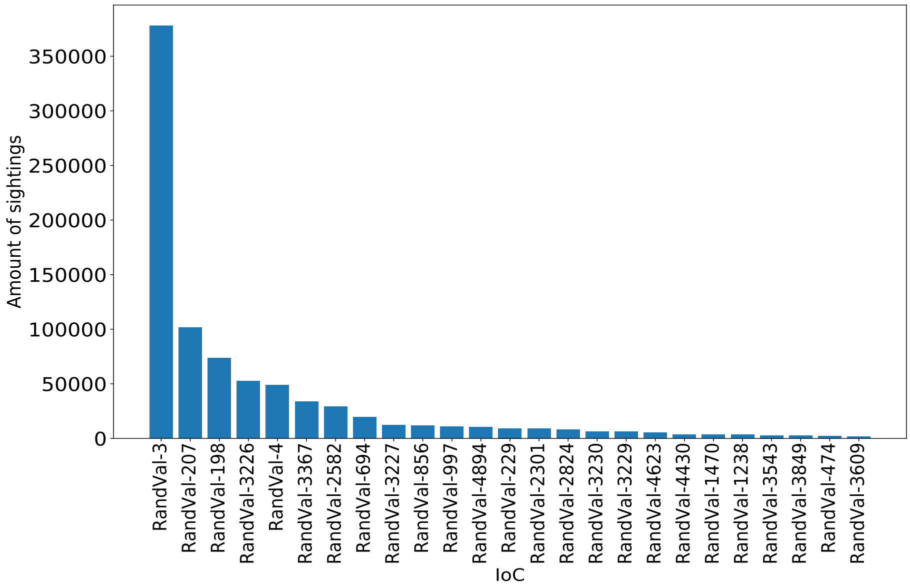

Figure 6 – Distribution of sightings per IoC

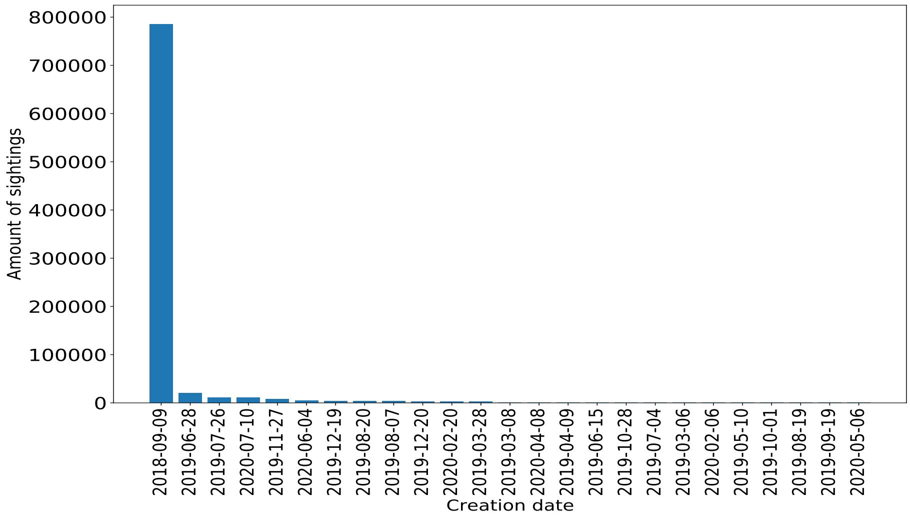

Figure 7 – Distribution of sightings by creation date

Additionally, it is worth noting that 533 IoCs were created on this day, around 9.2% of all IoCs available (see Figure 8). This indicates a Pareto distribution where 9.2% of IoCs created in a certain date are responsible for 88.02% of sightings.

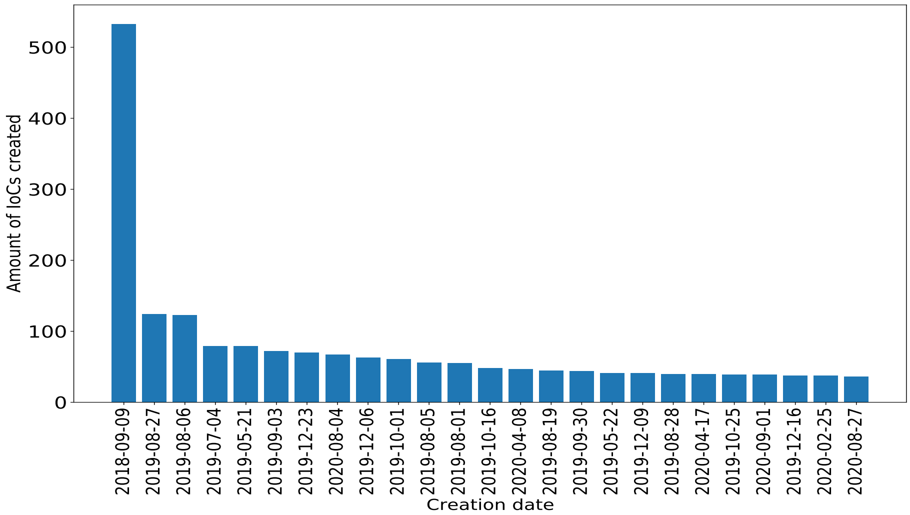

Figure 8 – Distribution of IoCs created by date

Nearly 80% of all first sightings occur within the first 10 days after its IoC is created. Even so, we have observed that there is an almost even distribution of the remaining first sightings throughout the monitoring period (Figure 9), with the largest period between an IoC's creation date and its first sighting spanning the 2 years of trace.

After personal communication with the group that provided the dataset, we learned that all IoCs created on or before 9/9/2018 were marked as being created on 9/9/2018. In future work, we plan to take this information into account when analyzing our data, e.g., by accounting for censored data using tools from survival analysis.

### 3.2.3 It may take long for the first sighting to occur, but once it occurs the sightings are typically concentrated

The time between sightings is typically shorter than the time between creation date and first sighting.

Figure 10 gives us a few insights about the distance between creation date and first sighting. The immediate takeaway is that 50% of all IoCs have their first sighting within its first 24 hours after being created. Also, since an IDS can produce sightings based on previous logs (IKLODY et al., 2018), we see that a portion of sightings happens before the creation of the IoC ($timedelta = -1$). The remaining segments in the chart reinforce the need to face the problem through the lenses of an aging model, as the density of first sightings in a day greatly drops, particularly after the 10 days mark.

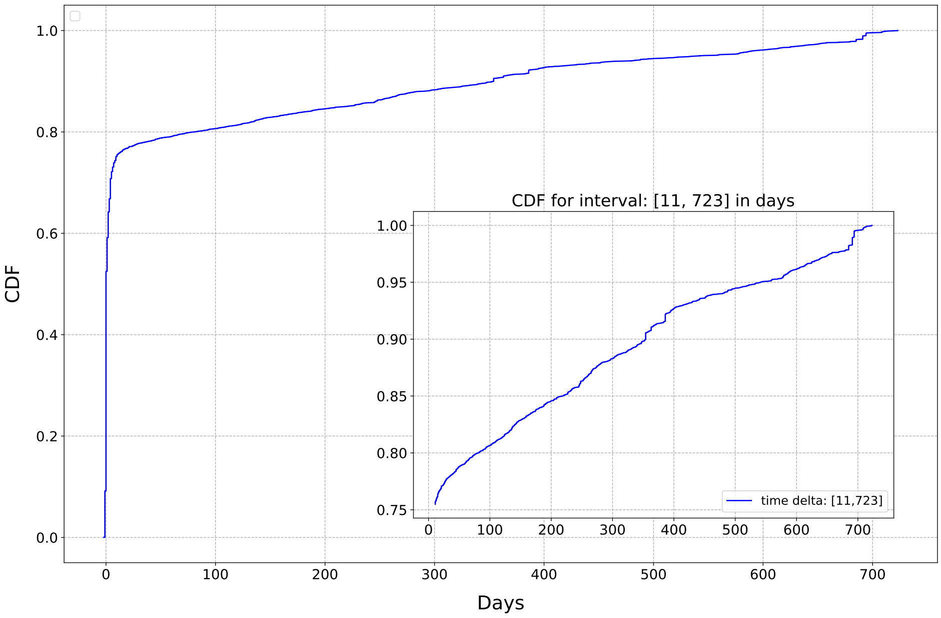

Figure 9 – External plot: CDF of time between first sighting and creation date. Internal plot: Zoomed in look at the external CDF.

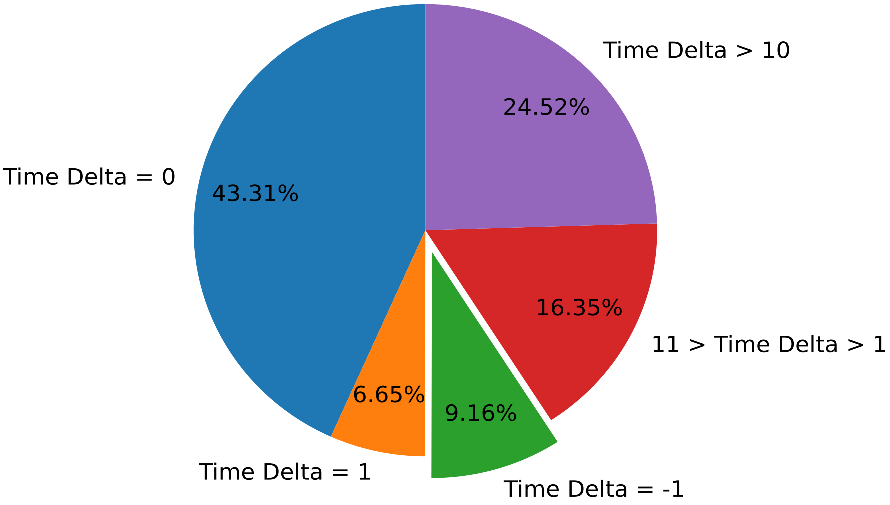

Figure 10 – Distribution of distance between creation date and first sighting.

## 3.3 CATEGORIES OF IOCS

We then separate data by its category. By doing this, we can explore trends specific to each category and allow a deeper understanding of its separate behavior.

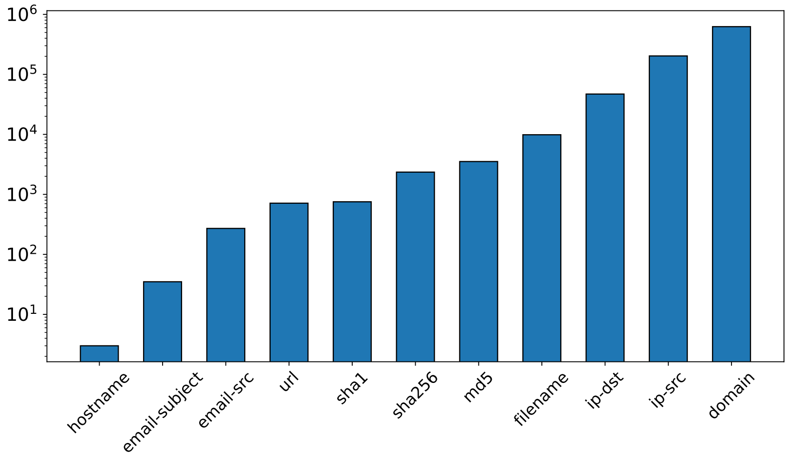

Figure 11 – Amount of sightings per IoC type, in logarithmic scale.

### 3.3.1 IP and domain-related IoCs are the most frequent

The first insight provided by this process of categorization on the trace is revealed by Figure 11, in which is possible to compare the presence of each category on the trace. The overwhelming disparity between the amount of sightings of type `domain` and of type `hostname`, by multiple orders of magnitude, may be caused by an intrinsic difference in quality for IoCs of these categories. Alternatively, if there are more indicators of a certain type being monitored by the entities that share information with the entity monitoring and the entity itself, then there will be more sightings of that specific category appearing in the trace as well, thus forming a potential bias.

Although Figure 11 may indicate that `domain` IoCs are the most relevant when looking only at the amount of sightings produced, Figure 12 shows that such trend was not present in 2020, as the amount of `domain` sightings greatly decreased.

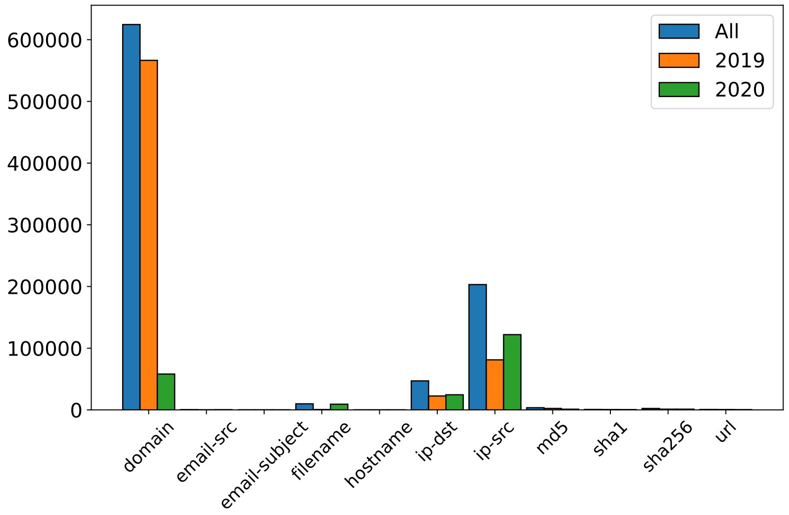

Figure 12 – Subdivision of the number of sightings per category into different years.

### 3.3.2 IP and domain-related IoCs are more ephemeral than their hash-related counterparts

It is also possible to apply categorization to the analysis of time between sightings seen in Figure 3, by considering only the intervals of time (also referred to as Time Delta) between sightings pertaining to IoCs of same type (see Figure 2), a cumulative distribution function (CDF) curve can be plotted for each category as shown in Figure 13.

The plot in Figure 13 reveals that the vast majority of sightings within a same category occur in relatively very short time span, however, some sightings may occur up to 8 months after this initial burst of sightings. It is also noticeable that `hostname` and `email-sbj`, due to a small sample of sightings in the trace (see Figure 11), present themselves as outliers to other categories.

Figure 14 further depicts the behavior of the curves for time between sightings, exhibiting the cumulative probabilities for time deltas up to 30 days and 14 days in each category. These plots, however, present significant distinctions even for categories that previously appeared to behave in similar manner, such as `filename` and `md5`.

Since different types of IoC have different associated behavior, regrouping them into coarse groups can provide further insights into types with similar characteristics and also simplify our categorization. However, we need to consider outlier categories that could join a group and skew its data, such as `hostname` and `email-sbj`.

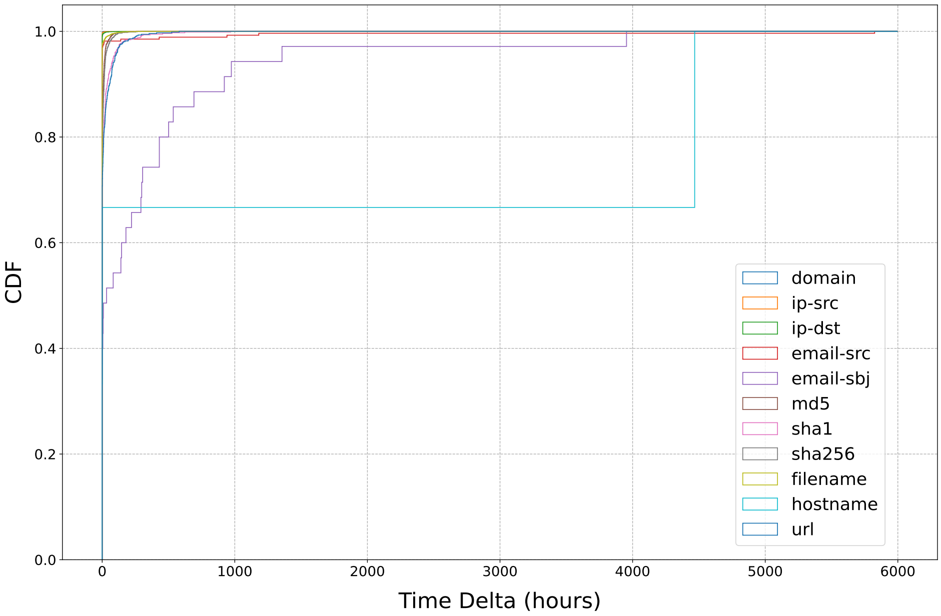

Figure 13 – CDF curves representing the time between sightings for each category.

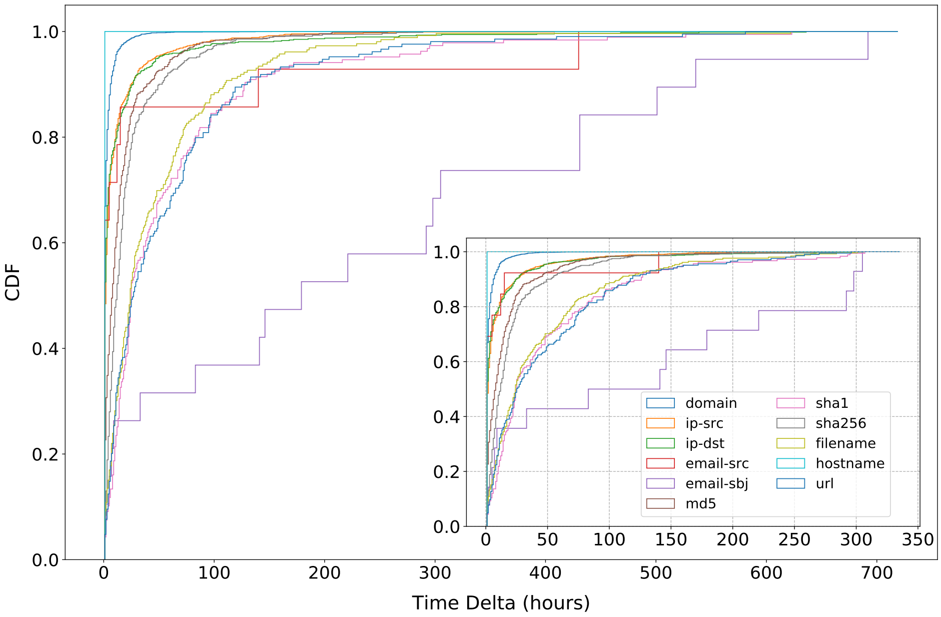

Figure 14 – External plot: CDF curves for intervals between sightings up to 30 days. This is the internal plot of figure 13.
Internal plot: CDFs of interavals ranging from 1 hour to 14 days (or 336 hours).

### 3.3.3 Grouping IP-related and hash-related IoCs provides further insight on their dynamics

In Figure 15, we present the CDF formed by the inter-sighting times for each IoC in each category for each of the following groups:

- IP: `ip-src + ip-dst`;
- Host: `domain + url + hostname`;
- Mail: `email-subject + email-dst`;
- Hashes: `md5 + sha1 + sha256`;
- Filename: `filename`.

Further decreasing the granularity of the CDFs, Figure 16 shows that it is possible to create an even more simplified analysis. In contrast, this may incur susceptibility to misjudgment of risks.

When analyzing the time between sightings of the same IoC, the vast majority of all intervals were short, rarely exceeding a week in time and with very few intervals getting close to the biggest time delta of 461 days. However, when analyzing only the intervals of time between a creation date and first sighting, we observe a much higher average of 83 days and a maximum interval of 722 days.

Furthermore, applying the same process of categorization to the trace, Figure 17 reveals that not only intervals reach much longer periods of nearly 2 years, but also, some

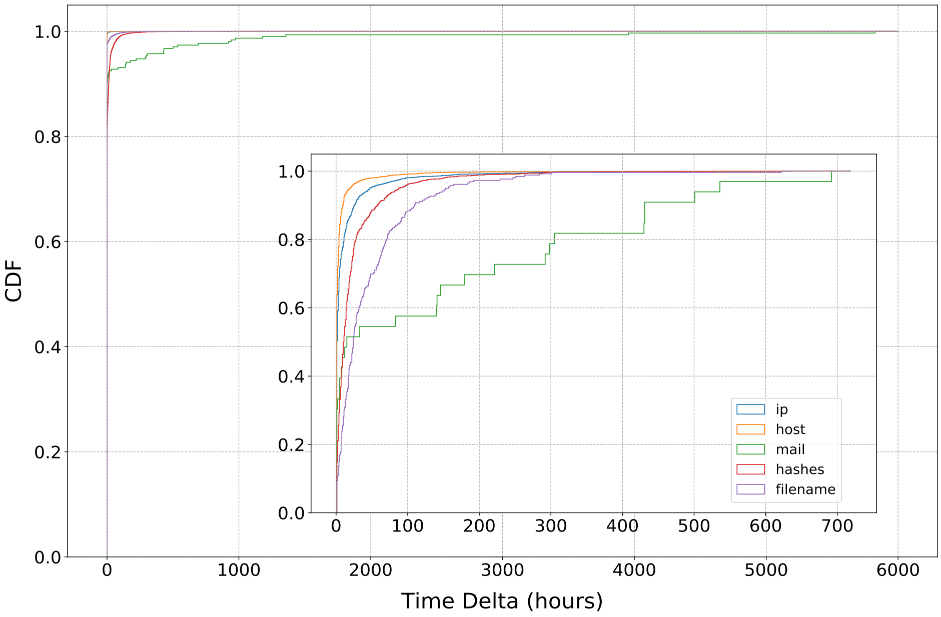

Figure 15 – External plot: CDFs of time between sightings for groups of categories. Internal plot: CDFs of time between sightings for the same groups, limited to time deltas pertaining to the interval [0,720], i.e., up to 30 days.

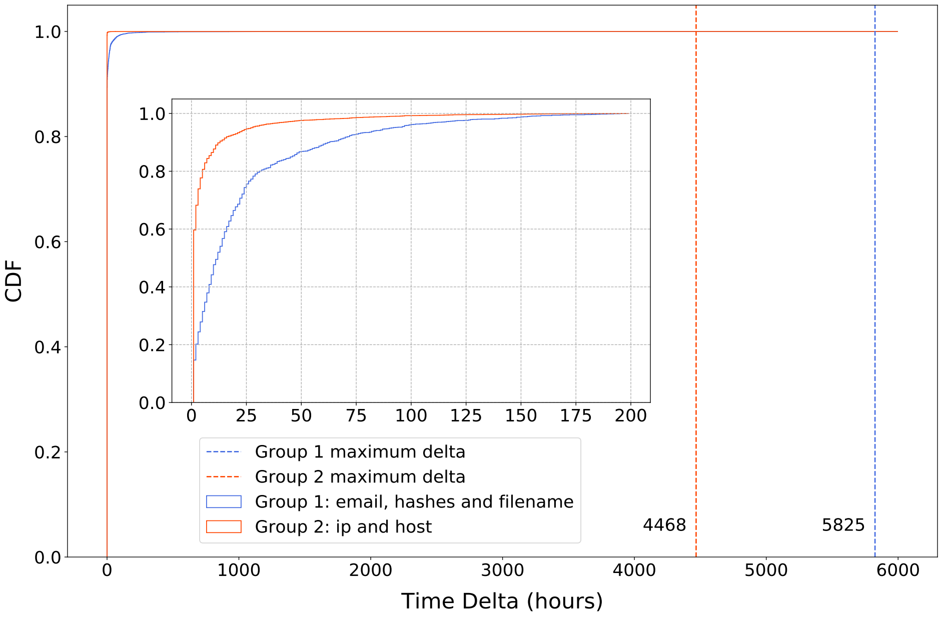

Figure 16 – External plot: The CDFs for the coarse groups: (`email + hashes + filename`) and (`ip + host`).
Internal plot: CDFs taking into consideration only the time deltas up to 200 hours.

categories have more than 80% of its time deltas between creation dates and first sightings distributed almost evenly starting a few hours after its creation to 17,375 hours (1.98 years).

Again, by merging groups of categories together, Figure 18 shows that even utilizing the same criteria used for Figure 16, the general behavior of time between creation date and first sighting vastly differs from the variation between sightings only. Considering this clear distinction, we consider the period between a creation date and its first sightings and the period of observation after the first sighting as separate phases.

Such disparities can impose problems when considering a TTL model (see Chapter 4), as coming up with a single TTL that describes both phases will likely produce either a large amount of misses for first sightings or IoCs monitored for too long. Therefore, as a natural expansion of our work in Chapter 4, a multi-TTL model is envisioned for future work (see Chapter 5.1).

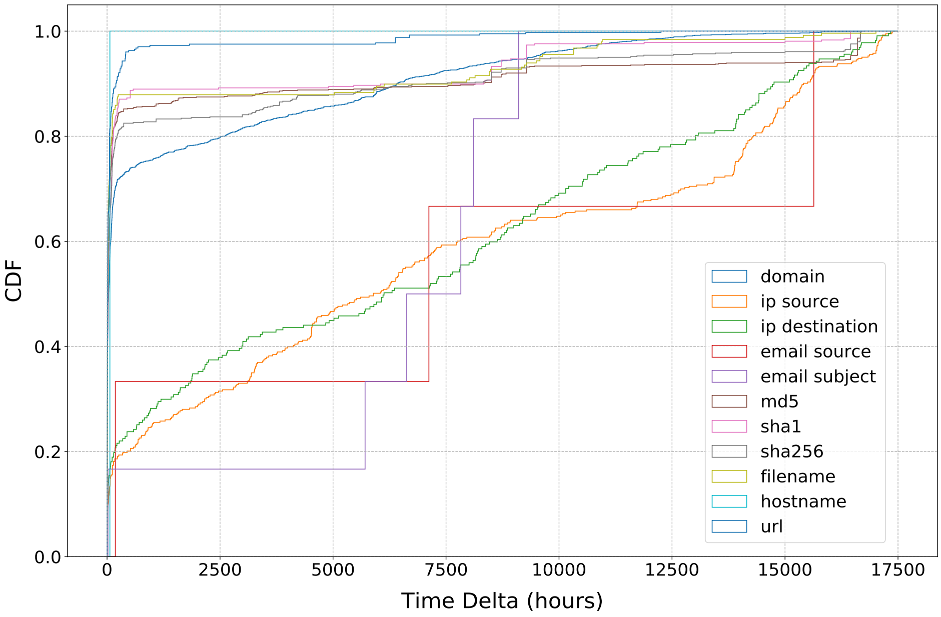

Figure 17 – First sightings to creation date of IoCs separated by category (analog to figure 9).

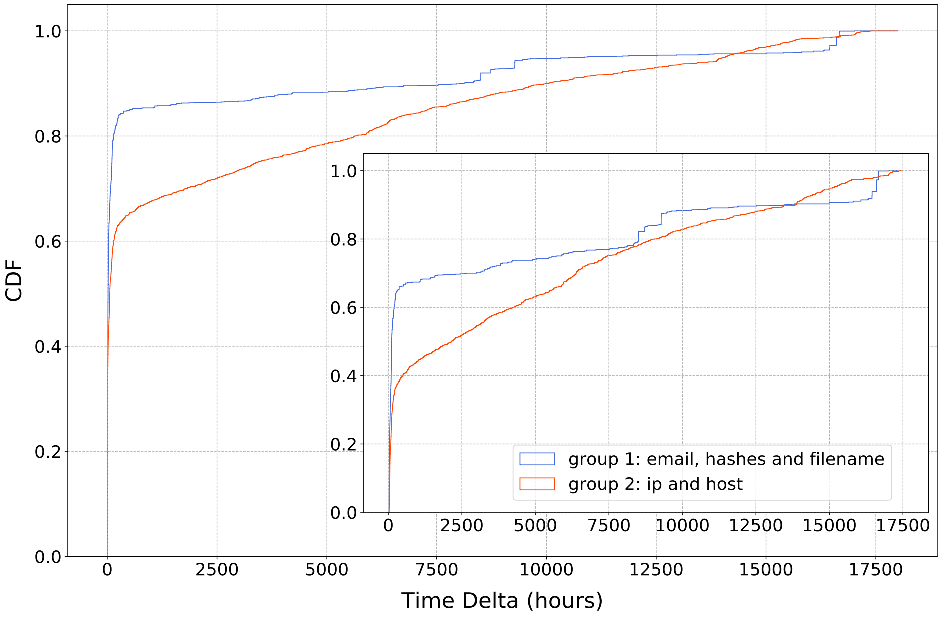

Figure 18 – External Plot: Grouping for intervals between an IoC's first sighting and creation date.
Internal plot: Analysis of the external CDFs restricted to the interval [50,17500].

## 3.4 ANALYSIS OVER EVENTS

In previous sections, we analyzed sightings relative to its IoC and IoC type. As another perspective for characterizing the trace, we consider sightings relative to events by aggregating all sightings of IoCs in the same event. Recall that each row in our trace consists of a sighting of an IoC, and such sighting also contains the identifier of the event that produced the sighting. Also, remember that event is a logical group of IoCs related to the same campaign. In this section, we analyze the behavior of events and their general relationship.

### 3.4.1 A few events concentrate most of the sightings

Our trace contains 2,814 events. Figure 19 shows the CDF of the amount of sightings per event. Even though more than 90% of events have very few sightings associated with them, the minority of events concentrates thousands of sightings. Observing the internal plot, we can infer that once an event has gathered more than a few hundred sightings, then there is a reasonably high probability it will reach multiple thousands of sightings. Although the amount of events with more than 80,000 sightings is very scarce, the figure indicates that once the 20,000 sightings mark has been reached, it is likely the event will have many more sightings.

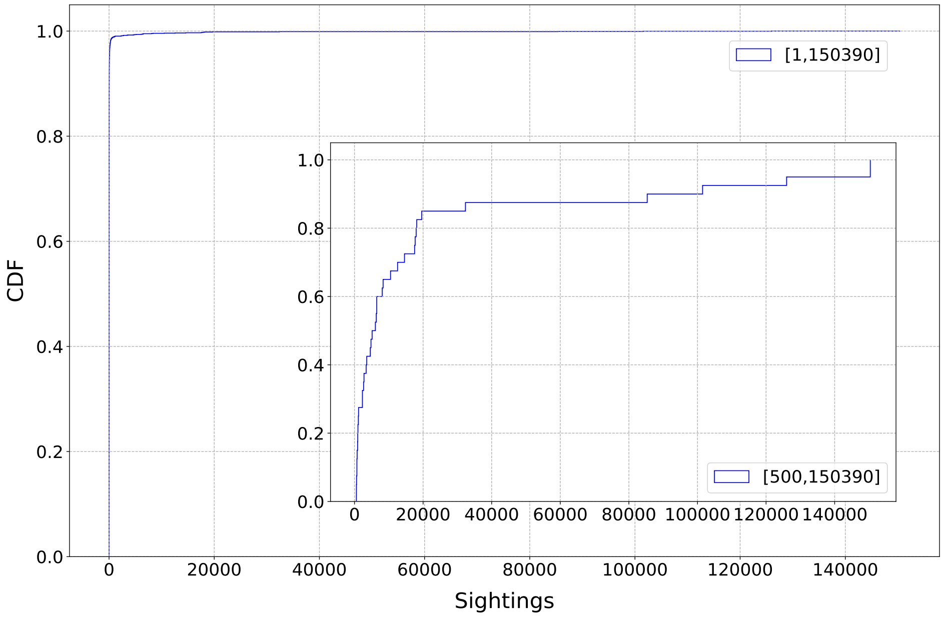

Figure 19 – External plot: CDF of the amount of sightings per event, ranging from 1 sighting to 150,390 sightings in an event.
Internal plot: A CDF of the restricted interval [500,150390], i.e., observing only the behavior of events that have more than 500 sightings.

Table 4 – Correlation matrix for categories

| | **domain** | **ip-src** | **ip-dst** | **email-src** | **email-sbj** | **md5** | **sha1** | **sha256** | **filename** | **hostname** | **url** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **domain** | 100.0% | 40.44% | 0.07% | 0.0% | 0.04% | 0.17% | 0.13% | 0.13% | 0.09% | 0.0% | 1.82% |
| **ip-src** | 51.57% | 100.0% | 0.01% | 0.0% | 8.85% | 0.31% | 0.0% | 0.15% | 0.12% | 0.0% | 0.01% |
| **ip-dst** | 89.51% | 0.01% | 100.0% | 0.0% | 0.0% | 69.63% | 69.74% | 69.54% | 0.03% | 0.02% | 0.02% |
| **email-src** | 0.0% | 72.69% | 0.0% | 100.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |
| **email-sbj** | 48.57% | 31.43% | 0.0% | 0.0% | 100.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% |
| **md5** | 13.15% | 2.04% | 6.49% | 0.0% | 0.0% | 100.0% | 33.47% | 46.47% | 12.78% | 0.0% | 3.03% |
| **sha1** | 20.72% | 0.8% | 7.97% | 0.0% | 0.0% | 95.22% | 100.0% | 94.16% | 13.01% | 0.0% | 7.57% |
| **sha256** | 17.39% | 1.23% | 6.89% | 0.0% | 0.0% | 65.01% | 45.88% | 100.0% | 12.03% | 0.0% | 4.12% |
| **filename** | 1.55% | 0.51% | 0.05% | 0.0% | 0.0% | 4.41% | 1.58% | 2.63% | 100.0% | 0.03% | 0.51% |
| **hostname** | 0.0% | 0.0% | 100.0% | 0.0% | 0.0% | 0.0% | 0.0% | 0.0% | 100.0% | 100.0% | 0.0% |
| **url** | 42.44% | 2.52% | 0.28% | 0.0% | 0.0% | 8.12% | 6.44% | 5.88% | 2.94% | 0.0% | 100.0% |

### 3.4.2 IPs and domain-related IoCs tend to be jointly sighted, the same holding for hash-related IoCs

By combining the ideas of analyzing the trace by looking at categories and by looking at events, we have produced a correlation matrix (see Table 4) in which we can observe the relationship between types. In the table, for a row of type $X$, a column of type $Y$ represents the percentage of sightings from IoCs of type $X$ that are present in events that contain at least one sighting for IoCs of type $Y$. Diagonal cells correspond to the total amount of sightings for each category in the trace, as 100% of all sightings of type $X$ appear in events with a type $Y$, for $X = Y$ in this case.

From the table, we see that 89.51% of all `ip-dst` sightings appear in events that also contain at least one sighting for IoCs in the `domain` category. Also, IoCs of type `sha1` have 95.22% and 94.16% of their sightings in events that also contain sightings for `md5` and `sha256`, respectively. Furthermore, decisions regarding the security of the system can be made taking different considerations based on such a correlation matrix.

# 4 MODEL

In this chapter, we describe models developed to reduce false positive IoCs and the monitoring cost associated with having too many IoCs in the system. We start with a description of the methodology used, then show some basic statistics of our trace, followed by simple boundary values and a first model to output the best TTL given a target hit ratio. Then, we consider categories as a single flow as a possible venue. In addition, we introduce utilities and cost to assist our second model: given a cost associated with monitoring an IoC and a cost associated with missing a sighting, we provide the TTL where the total cost associated is minimized. Lastly, we provide a third model, where no input beyond the trace is required, and the output is in the form of two thresholds, where below or above them indicates a policy of always or never monitoring should be considered.

## 4.1 IOC DECAYING MODEL

To each IoC we associate a corresponding time-to-live (TTL). The TTL is initialized to a constant value, and is decremented at every time unit. When TTL reaches zero, the corresponding IoC monitoring is discontinued. Such TTL decaying model has a number of different flavors, e.g., varying with respect to how it reacts to sightings. Under TTL with reset, the TTL is reset to its initial value whenever a sighting occurs. Under TTL without reset, in contrast, sightings do not impact the TTL dynamics. In any case, note that the TTL dynamics are decoupled across multiple IoCs. We decided to use TTL models at first due to their compared simplicity and literature available.

The rate at which TTL decays over time, as well as the initial value of the TTL, are two among the many parameters that can be tuned according to user needs. In what follows, we focus on the latter, assuming a linear decay of TTL over time. We let $T$ denote the initial TTL.

## 4.2 METHODOLOGY

Let the coverage, or hit ratio, of a parameterized TTL model be the fraction of IoC sightings that occur while the corresponding IoC is being monitored. A sighting to an IoC whose TTL value equals zero is said to be uncovered, contributing towards its miss ratio. Correspondingly, the monitoring cost at any given point in time is proportional to the number of IoCs whose TTL value is greater than zero.

## 4.3 SIMPLE DETERMINISTIC BOUNDS

We begin by estimating an upper bound on the TTL value to cover all sightings for all IoCs. Indeed, a conservative approach towards IoC monitoring consists in setting TTL to a large enough value that, in retrospect, would have covered all sightings. That approach is optimal if monitoring costs are negligible, e.g., if the team responsible for monitoring IoCs is large enough to treat all alarms in an accurate and timely manner.

In our trace, the largest gap between the first and last sightings towards an IoC equals 461 days. Following this conservative monitoring strategy, i.e., letting $T = 461$, and assuming that IoCs are created at a rate of $\lambda$ IoCs per time unit, the expected number of IoCs to be monitored at any point in time equals $461 \cdot \lambda$. This estimate, however, is very sensitive to outliers. In addition, new outliers may cause the gap between first and last sighting to increase over time, motivating the use of statistical tools to parameterize TTL in order to determine when and if IoCs should be evicted.

## 4.4 STATISTICAL ANALYSIS

To cope with outliers and with the need to allow a certain level of missed sightings, we consider statistical approaches to parametrize TTLs. In the simplest setting, we take as inputs the target hit ratio $t$ (with corresponding miss ratio $1 - t$) and the cumulative distribution function (CDF) of the time between consecutive sightings, $F(x) = P(X < x)$, where $X$ is a sample from the distribution of the time between sightings. Then, we let $T = F^{-1}(t)$.

For large values of $t$, this model clearly degenerates to the simple deterministic bound discussed in the previous paragraph. Smaller values of $t$ allow us to trade-off between coverage and monitoring costs. In our trace, to capture 90% of sightings of IoCs related to emails, we must let $T = 38$ days. In this case, a 10% reduction in coverage corresponds to a 95% decrease in monitoring costs.

## 4.5 ACCOUNTING FOR CATEGORIES

IoCs can be categorized according to different criteria. In the previous example, we illustrated how IoC types can be instrumental to set TTLs and decrease monitoring costs. In our trace we count with eleven IoC types: `md5`, `sha1`, `sha256`, `ip-src`, `ip-dst`, `email-subject`, `email-dst`, `domain`, `hostname`, `filename` and `url`. Conditioning TTL values to IoC types allows us to significantly reduce the impact of outliers, which skew the TTL values for the whole trace but may not impact certain categories. Alternatively, categories may also be instrumental to parametrize the TTL model with reset, taking the set of sightings towards each IoC category as a single stream. In that case, sightings are distinguished by their IoC type, but not by their IoC identifiers. The maximum time between sightings through the whole trace, reported above as 461 days, is drastically reduced to 243 days when accounting for sightings of IoCs of the same type.

The categories discussed above can be split or grouped. As an example, the eleven categories may be grouped into five coarser groups, briefly discussed in Section 3.3.2: hashes (`md5`, `sha1`, `sha256`), ips (`ip-src`, `ip-dst`), email (`email-subject`, `email-dst`), host (`domain`, `hostname`, `url`) and `filename`. The coarser the granularity, the simpler the model, requiring less parameters to be evaluated, but the more sensitive are each of the classes to outliers. Alternatively, additional features may be available to produce supervised or unsupervised clusters of IoCs to be treated in an integrated fashion.

## 4.6 BASIC STATISTICS

Table 5 provides a few basic statistics derived from the considered dataset together with the TTL models. Basically, it is separated into three categories. First, TTL with reset, meaning that whenever a new sighting occurs, the TTL is reset to $T$. In this case, we calculate four main values for the dataset considering only sightings or sightings and creation dates together. The first row, $R_L$, represents the lower threshold. Let the ratio between monitoring cost and miss cost be $r$. If $r < R_L$, then the best alternative is to always monitor every IoC because the cost of monitoring is too small compared to the cost of miss – or the cost of miss is too high compared to the cost of monitoring. Similarly, $R_U$ represents the upper threshold where $r > R_U$ means we should never monitor, i.e. the cost of monitoring is too high compared to the cost of miss. Then, we highlight $\hat{T}$, the maximum distance between adjacent occurrences. Exemplifying, if considering only sightings, this is the maximum number of days between two consecutive sightings in our dataset, which is the maximum TTL required to cover every sighting. $\bar{T}$ is the mean time between those adjacent occurrences, which tells us the rate of new sightings.

We then describe TTL without reset, an approach where a TTL is fixed and a new sighting has no effect. When an IoC enters the system, we define an eviction date and remove it by then. In addition to $R_L$ and $R_U$ seen above, we also show $\dot{T}$, equivalent to the maximum TTL required to not miss any sighting, similar to $\hat{T}$ for TTL with reset.

Ultimately, we describe a few general information about the dataset, such as total number of entries in the considered dataset, IoCs with a first date before creation date (see Section 3.2.3), IoCs with no creation date available, and relevant dates.

Also, as an usage example of the table, considering only sightings, say that the ratio of monitoring cost per IoC per day divided by the cost of a miss, both in dollars, is greater than 1/312, then it is beneficial to monitor all IoCs forever.

Table 5 – Thresholds on the monitoring cost over miss cost ratio and maximum time between consecutive events.

| | Only sightings | Sightings and creation dates |
|---|---|---|
| TTL with reset | | |
| $R_L$ (lower threshold) | 1:476,197,152 | 1:28,012,293 |
| monitoring cost:miss cost motivating always monitor | | |
| $R_U$ (upper threshold) | 1:312 | 1:2,152 |
| monitoring cost:miss cost motivating never monitor | | |
| $\hat{T}$ (maximum distance btw. adjacent occurrences of sightings and creations per IoC) | 461 days | 722 days |
| $\bar{T}$ (mean distance between adjacent occurrences of sightings and creations per IoC) | 0.1 day | (83 days between creation and 1st sight.)<br>0.6 day |
| TTL without reset | | |
| $R_L$ (lower threshold) | 1:47,620,165 | 1:843,470 |
| $R_U$ (upper threshold) | 1:771 | 1:2115 |
| $\dot{T}$ (max. dist. btw. first and last occurrences per IoC) | 496 days | 722 days |
| general statistics | | |
| # sightings+creations | 892,240 | 898,026 |
| # IoCs with first sighting before creation date | 530 IoCs | |
| # IoCs with sighting | 5,789 | |
| # IoCs without creation date | 3 | |
| # IoCs (total) | $\approx 14,000,000$ | |
| total time | 724 days | |
| first creation date | 2018-09-09 | |
| last creation date | 2020-09-02 | |
| first sighting | 2019-04-24 | |
| last sighting | 2020-09-02 | |

## 4.7 UTILITIES AND COSTS

Next, we consider the availability of information about monitoring costs and costs associated with missing a sighting, to determine the target hit ratio $t$. Together, such costs can assist in the flexible monitoring of IoCs. Such costs, e.g., measured in dollars per time unit and dollars per missed sighting, respectively, can be used to establish a utility function that impacts TTL values. This is in contrast to setting TTL values directly based on the fraction of missed sightings that can be tolerated.

Introducing monetary costs may impose additional challenges, as determining such costs is non-trivial. However, monetary costs may help convey the role of the IoC aging model in the considered organization, bridging the gap between IoC monitoring strategies and other elements of the business workflow. Monetary costs can be determined exogenously based on related literature or on information provided by certain products, such Azure Sentinel Threat Intelligence (ASTI). ASTI offers two price models: Capacity Reservations and Pay-As-You-Go. In the latter, the current cost is USD 2.46 per GB-ingested.

Alternatively, the ratio between monitoring and missing costs can be estimated in an endogenous fashion, using data from the collected traces. Indeed, the trace of sightings implies that if the cost ratio is above a certain threshold, one should never monitor any IoC (no-monitoring extreme). At the other extreme of the spectrum, when the ratio is below a lower threshold all IoCs should be constantly monitored (always-monitoring extreme). Knowing such two thresholds, and understanding how the cost ratio impacts monitoring strategies, together with historical information about monitoring practices in a given business, provides insights on the current and prospective target cost ratios.

To find the two thresholds referred to in the above paragraph, we define TTLs ranging between 0 and the maximum interval between sightings (see Section 4.3). For each TTL value we compute, in retrospect, using the provided trace, the corresponding monitoring and missing costs. The monitoring cost is the number of days we monitor each IoC in our system multiplied by the cost of each day of monitoring. The missing cost is the number of missed sightings multiplied by the cost of each miss.

Let $C$ be the total cost, and $C_M$ and $C_S$ be the monitoring and missed sighting costs, respectively. Under the above simple model, the total cost is a linear function of the time that IoCs were monitored (must also consider IoCs with no sighting) and the number of missed sightings:

$$\begin{aligned} C(M, S; C_M, C_S) &= C_M \sum_{i=1}^{I} M_i + C_S \sum_{i=1}^{I} S_i && (4.1) \\ &= C_M M + C_S S && (4.2) \end{aligned}$$

where $M_i$ and $S_i$ are the monitoring time and number of missed sightings for the $i$-th IoC, and $M$ and $S$ are the corresponding quantities accounting for all IoCs. Note that $M$ and $S$ are functions of $T$. As the dependency of $M$ and $S$ on $T$ may be non-trivial, e.g., non-convex, we proceed with a trace-driven exploration to 1) determine the best TTL value, given the costs of monitoring and missing and 2) search for the two cost ratios that correspond to the extremal thresholds discussed above. The optimization problem corresponding to the optimal TTL estimation is given as follows:

$$\begin{aligned} T^* : \quad & \text{Argmin}_T & C(M, S; C_M, C_S) \\ & \text{Subject to} & M = g(T) \\ & & S = h(T) \end{aligned}$$

Let $R = C_M/C_S$. The optimization problem corresponding to the extremal cost ratio estimation, to determine an upper bound on the cost ratio $R_U$ beyond which IoCs should never be monitored, is given as follows:

$$\begin{aligned} R_U :&\text{Min} \quad R \\ &\text{Subject to} \\ &\quad (\text{Argmin}_T R \cdot g(T) + h(T)) = 0 \end{aligned}$$

Note that $g(0) = 0$, therefore for $R > R_U$ we have $C = C_S h(T)$. In the regime wherein monitoring costs are high, no IoCs are monitored and the ultimate cost depends only on the number of missed sightings.

Correspondingly, the optimization problem to the determine a lower bound on the cost ratio $R_L$ below which IoCs should always be monitored is given as follows:

$$\begin{aligned} R_L :&\text{Max} \quad R \\ &\text{Subject to} \\ &\quad (\text{Argmin}_T R \cdot g(T) + h(T)) = \widetilde{T} \end{aligned}$$

where $\widetilde{T}$ is the maximum feasible TTL value. Under a trace-driven approach, $\widetilde{T}$ can be set as the maximum interval between sightings (see Section 4.3). Assuming $h(\widetilde{T}) = 0$, i.e., no sightings are missed when $T$ is set to $\widetilde{T}$, condition $R < R_L$ implies that $C/C_S = R \cdot g(T)$. In the regime wherein monitoring costs are low, all IoCs are monitored and the ultimate cost depends only on the product $C_M \cdot g(T)$.

Note that in the above formulation we assumed that functions $g(\cdot)$ and $h(\cdot)$ are obtained directly from traces. Alternatively, we envision that approximating those functions through simple expressions may be instrumental to express the solutions to the above problems in closed-form, which we leave as subject for future works.

# 5 CONCLUSION

In this work we leverage a trace of real world IoCs and its corresponding sightings to create a TTL model where the importance of an IoC decreases over time. With the characterization of the dataset, we found that only around 0.04% of IoCs have an associated sighting. Out of the 5,789 IoCs with an associated sighting, 3,154 (54.48%) only have one sighting. Nevertheless, we define how much time an IoC should be kept in the system with a TTL approach. We begin by considering a simple deterministic bound, corresponding to the maximum time between sightings found in our trace. Under this model, every IoC in the system should be monitored for roughly two years. In these TTL models, we define a time-to-live to an IoC and, using the trace, calculate which percentage of the IoCs are covered, achieving a target ratio $t$. By leveraging different categories composed of multiple IoC types, we are able to isolate outliers from skewing the TTL across categories, and reduce the TTL by roughly 90% accounting for a miss ratio of up to 10%. Additionally, given monitoring and miss costs, we evaluate which TTL yields an optimal total cost. Still, if neither a percentage of acceptable amount of sightings lost is given, nor related costs, we then provide two thresholds. If the ratio between cost of monitoring and miss is smaller than the lower threshold, always monitor, i.e. the cost of monitoring is too small compared to missing. Similarly, if such ratio is above the upper threshold, never monitor is the best alternative. Anything in between is up to a trade-off decision. Such thresholds pave the way towards a principled understanding of how business models, involving monetary costs, can impact the nuts-and-bolts of the operations of a SOC, involving the assignment of TTLs towards IoCs.

It is also important to note that the results presented in this work are intrinsically dependent of the trace used. However, we believe the framework employed and the results found can be generalized to other environments.

## 5.1 FUTURE WORK

Despite the results found, we believe there is room for improvement on the TTL model by considering a multi-TTL approach, where an IoC may go through different phases. As seen in Chapter 2, for example, the time between an IoC creation and its first sighting is usually larger than the time between its first and second sighting, so considering a larger TTL in the first interval and a smaller in the second can be beneficial.

Moreover, an approach to fit the CDFs presented as a known distribution can help generalize the findings in our work for future research, e.g. fitting the distribution of sightings in a Pareto distribution. In future research, it is also possible to explore the usage of specific tags and Natural Language Processing (NLP) in the textual description of

IoCs to achieve a more refined TTL model. In such case, we could leverage the description of an IoC and the context provided by tags to assist in the TTL assigned, e.g. IoCs related to APT activities should use a higher cost of missing and a larger TTL. However, these approaches require a non-anonymized dataset, which is not currently possible in our case.

# REFERENCES

AbuseIPDB. **AbuseIPDB**. 2021. Disponível em: https://www.abuseipdb.com/.

ALVES, F. et al. Processing tweets for cybersecurity threat awareness. **Information Systems**, Elsevier, v. 95, p. 101586, 2021.

Binary Defense. **Binary defense banlist**. 2021. Disponível em: https://www.binarydefense.com/banlist.txt.

BOUWMAN, X. et al. A different cup of TI? the added value of commercial threat intelligence. In: **29th USENIX Security Symposium (USENIX Security 20)**. [S.l.: s.n.], 2020. p. 433–450.

BOUWMAN, X. et al. Helping hands: Measuring the impact of a large threat intelligence sharing community. In: **31st USENIX Security Symposium (USENIX Security 22)**. Boston, MA: USENIX Association, 2022. Disponível em: https://www.usenix.org/conference/usenixsecurity22/presentation/bouwman.

CARRA, D.; NEGLIA, G.; MICHIARDI, P. Ttl-based cloud caches. In: IEEE. **IEEE INFOCOM 2019-IEEE Conference on Computer Communications**. [S.l.], 2019. p. 685–693.

CATAKOGLU, O.; BALDUZZI, M.; BALZAROTTI, D. Automatic extraction of indicators of compromise for web applications. In: **Proceedings of the 25th international conference on world wide web**. [S.l.: s.n.], 2016. p. 333–343.

CONSULTING, B. **CC tracker**. 2021. Disponível em: https://osint.bambenekconsulting.com/feeds/c2-ipmasterlist.txt.

CYBERCURE. **Cybercure free intelligence feed**. 2021. Disponível em: https://api.cybercure.ai/feed/get_ips.

DIONÍSIO, N. et al. Cyberthreat detection from twitter using deep neural networks. In: IEEE. **2019 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2019. p. 1–8.

ERMERINS, J. et al. Scoring model for iocs by combining open intelligence feeds to reduce false positives. 2020. https://www.os3.nl/_media/2019-2020/courses/rp1/p55_report.pdf.

FAGIN, R. Asymptotic miss ratios over independent references. **Journal of Computer and System Sciences**, Elsevier, v. 14, n. 2, p. 222–250, 1977.

IKLODY, A. et al. Decaying indicators of compromise. **arXiv preprint arXiv:1803.11052**, 2018.

JUNG, J.; BERGER, A. W.; BALAKRISHNAN, H. Modeling ttl-based internet caches. In: IEEE. **IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428)**. [S.l.], 2003. v. 1, p. 417–426.

LE, B. D. et al. Gathering cyber threat intelligence from twitter using novelty classification. **arXiv preprint arXiv:1907.01755**, 2019.

LIAO, X. et al. Acing the ioc game: Toward automatic discovery and analysis of open-source cyber threat intelligence. In: **Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security**. [S.l.: s.n.], 2016. p. 755–766.

LONG, Z. et al. Collecting indicators of compromise from unstructured text of cybersecurity articles using neural-based sequence labelling. In: IEEE. **2019 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2019. p. 1–8.

MELL, P.; SCARFONE, K.; ROMANOSKY, S. Common vulnerability scoring system. **IEEE Security Privacy**, v. 4, n. 6, p. 85–89, 2006.

MICROSOFT. **Azure Sentinel pricing**: Current price of Azure's Sentinel service. [S.l.], 2021. Disponível em: https://azure.microsoft.com/en-us/pricing/details/azure-sentinel/. Acesso em: 27/10/2021.

MICROSOFT. **Azure Sentinel Threat Intelligence**: Threat indicators for cyber threat intelligence in Microsoft Sentinel. [S.l.], 2021. Disponível em: https://docs.microsoft.com/en-us/azure/architecture/example-scenario/data/sentinel-threat-intelligence. Acesso em: 27/10/2021.

MITTAL, S. et al. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In: IEEE. **2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)**. [S.l.], 2016. p. 860–867.

MOURA, G. C. et al. Cache me if you can: Effects of dns time-to-live. In: **Proceedings of the Internet Measurement Conference**. [S.l.: s.n.], 2019. p. 101–115.

SABOTTKE, C.; SUCIU, O.; DUMITRAș, T. Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In: **24th {USENIX} Security Symposium ({USENIX} Security 15)**. [S.l.: s.n.], 2015. p. 1041–1056.

WAGNER, C. et al. Misp: The design and implementation of a collaborative threat intelligence sharing platform. In: **Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security**. [S.l.: s.n.], 2016. p. 49–56.