University of Amsterdam

Security and Network Engineering

Scoring model for IoCs by combining open intelligence feeds to reduce false positives

February 9, 2020

Students:
Jelle Ermerins
jermerins@os3.nl

Niek van Noort
nnoort@os3.nl

Supervisors:
João de Novais Marques
Leandro Velasco

Abstract

Indicators of Compromise (IoCs) are used by organisations to identify possible threats. This research focuses on IP addresses, but could be applied on other IoCs like malware hashes and domain names. In this research a scoring model is designed with the aim of reducing the number of false positive IoCs. The scoring model uses multiple open threat intelligence feeds to determine a score for an IoC. This research shows how useful it is to combine different intelligence feeds, by looking at the quantity of overlapping IoCs, and independence between the feeds. Furthermore, the scoring model is time dependent by using a decay function. To determine if an intelligence feed can be trusted, a source confidence is calculated for each feed. This confidence is based on features like extensiveness, timeliness, completeness, and a whitelist overlap score. Finally, a final score gets calculated for an IoC by combining the scores from multiple feeds.

1 Introduction

Indicators of Compromise (IoCs) are used by organisations to identify possible threats. By identifying these threats, organisations are able to protect themselves and their customers. An IoC could for example be an IP address active in a botnet, a domain of a phishing site or a hash of malware files [1]. Collecting IoCs manually is intensive work, therefore automation methods have been invented, e.g. scraping blog posts to find IoCs with natural language processing [2]. The problem with these automated methods is the creation of false positives. An IP address could be wrongly accused of being malicious, or over time a malicious IP address could be re-allocated to a new owner [3].

To stay ahead of the threats, organizations need to share their collected IoCs. On the Internet several IoCs sharing sources can be found [1]. For example, Malware Information Sharing Platform (MISP) is a peer to peer platform where users can share their IoCs publicly or with a selected group of MISP users [4]. The problem with such a sharing platform is verifying if a sharing source of an IoC is trustworthy and does not create false positives.

In this research a scoring model is designed with the aim of reducing the number of false positive IoCs. This scoring model is created by combining different sources available to us [1]. In the context of IoCs these sources are called *intelligence feeds*. In our research, intelligence feeds are daily updated lists of sighted IoCs. Here, *sighted* means that an IoC was detected in malicious activities. We focus on intelligence feeds that provide malicious IP addresses, but the model is designed to also work with other types of IoCs, e.g. malware hashes. The scoring model will base the score of an IoC on the scores provided by the individual intelligence feeds themselves. The intelligence feeds base their score on features like the timestamp of when the IoC was last sighted and how many times it was sighted. Furthermore, the overlap of the intelligence feeds is calculated to verify that the feeds are independent from each other.

1.1 Related Work

To efficiently defend ourselves against threats on the Internet, sharing of IoCs is important. Previous research gives an overview of the MISP platform, used to share IoCs between organizations [4]. The focus of MISP is sharing IoCs, the overview defines different levels of sharing and explains the synchronization protocol of MISP. Furthermore, MISP is able to compute some statistics, such as the overlap of IoCs between sources. A feature of MISP, called sightings, lets MISP users label IoCs as true or false positives [5]. What lacks in MISP is a way to automatically evaluate the confidence we can put in a source and its IoCs to reduce the number of false positives.

In other previous research, a scoring model for IoCs within MISP communities was designed [3]. The scoring model consists of a base score and a decay rate that decreases the score of an IoC over time. By using a decaying score, an IoC can cease to exist when its malicious activities stop or produce a different IoC. The research focuses primarily on the decay rate of the scoring model. In continuing research, the scoring model is used to find the optimal parameters for the decay rate for a specific dataset of IoCs related to phishing campaigns [6]. In their experiments they show that with these parameters 40% of the IoCs are expired too early.

In 2019, a scoring model was introduced based on the source of the IoCs [7]. The score is determined by comparing the IoCs of the same and different sources, by looking at features derived from the Structure Threat Information Expression (STIX) standard. Some of the features introduced in this scoring model are: extensiveness (how much metadata is provided), completeness of source (how much the source has contributed to the total number

of IoCs), history of false positives, similarity of information between different sources, and timeliness (what source is the fastest with providing a IoC). The research only introduced a scoring model, but did not apply it nor investigate the usability and dependency of multiple open IoC sources in practice.

1.2 Research Question

The aim of this research is to design a scoring model for IoCs based on open intelligence feeds to reduce the number of false positive IoCs. The research question we want to answer is:

How can we use multiple open intelligence feeds in a scoring model to determine the quality of IoCs?

To answer the research question we want to know how much overlap different intelligence feeds have. When we look at the overlap of intelligence feeds, an important point is how independent the different feeds are from each other. Furthermore we want to make the scoring model time dependent, such that a score of an IoC decays over time. We also want to determine how much an open intelligence feed can be trusted, and how this integrates in the scoring model. Finally we are looking at how we calculate one score from multiple intelligence feeds with difference levels of trust. With these challenges of designing a scoring model in mind, the following sub questions need to be answered:

How independent are different intelligence feeds from each other?
How do we make the model time dependent?
How do we decide if we can trust an intelligence feed?
How do we calculate one score from multiple feeds with different levels of trust?

2 Methodology

The scoring model is used to determine the quality of an IoC to decide if an IoC can be considered as a false positive or not. An example of IoCs that are given as input to the scoring model could be IP addresses reported as malicious in the logs of a web server. Our scoring model consists of two important building blocks: the source confidence and the decay time. For each intelligence feed, one source confidence is calculated that predicts how much trust we can put in a feed. The source confidence depends on the quality of the IoC that a feed has delivered in the past and is explained in more detail in Section 2.4. A compromised host can be fixed, after which its malicious behavior stops. Therefore, we want our scoring model to be time dependent. In Section 2.3 we explain how to use the timestamp of the last sighting of an IoC to make its score decay over time. But first, we will take a look at the data that the scoring model depends on.

2.1 Collecting IoCs

Before we explain the design of our model, we need to know what data we can get from the intelligence feeds, and what data we can provide ourselves. The intelligence feeds that we use are AbuseIPDB [8], Binary Defense Banlist [9], C&C Tracker [10], and Cyber Cure free intelligence feeds [11]. A problem with multiple open intelligence feeds is that they don't provide the same data. Per IoC, some feeds only provide an IP address, others provide descriptions, timestamps, the number of sightings, or even a confidence score per IoC themselves. Timestamps are important to our model, because the decay time depends on timestamps. If a feed does not provide a timestamp, we can provide it ourselves. For each IoC we store the timestamp at which we have firstly and lastly seen it in an intelligence feed. Although these timestamps are less precise than when an intelligence feed provides its own

timestamps, decaying the IoC score over time will be possible for all intelligence feeds now.

To calculate the source confidence of the intelligence feeds, we need to keep track of the history of the IoCs provided by each intelligence feed. Therefore, we will access these intelligence feeds every day for a two week period to find new IoCs. We will collect the IoCs for each intelligence feed in a separate dataset. Per intelligence feed, for each IoC we will store general information about the IoC. This information consists of the IP address, a timestamp of the first time the IoC was added to the dataset and a timestamp of the last time we found this IoC in the particular intelligence feed. If intelligence feeds give us more data, this will also be stored for each IoC. This extra data is mostly important to calculate the extensiveness of the intelligence feeds, which is explained in Section 2.4 in more detail.

2.2 Calculating Independence and Overlap between Intelligence Feeds

After collecting IoCs, we need to determine the independence between the intelligence feeds. The independence between the feeds is important if the scoring model uses overlap between feeds as part of the score. If the intelligence feeds are dependent, two or more intelligence feeds can be seen as one feed with a higher weight, which is undesirable. The independence of feeds is determined by looking at how the feeds collect their IoCs. If the feeds are collecting IoCs from the same sources, these feeds are more dependent than the feeds that collect IoCs from different sources. A description for each intelligence feeds can be found in Appendix A.

Also the overlap between the different feeds is important to determine if it is useful to combine these feeds in the scoring model. If there is no overlap between the feeds, the score of each IoC would depend on only one feed. The overlap is calculated by looking if an IoC from one feed also exists in other feeds. This is done for every feed, resulting in a matrix containing the percentage of overlap between each pair of feeds. Equation 1 shows how to calculate the percentage of IoCs from $Feed_A$ that are also present in $Feed_B$.

$$Overlap = \frac{|Feed_A \cap Feed_B|}{|Feed_A|} \cdot 100 \tag{1}$$

When two feeds partly overlap, but the time of sighting of the overlapping IoCs have a large difference, it is unlikely that these sightings were caused by the same threat. In this case it is also unlikely that the feeds are dependent. If two feeds contain overlapping IoCs that are sighted within a small time period, it is more likely that these feeds are dependent. To distinguish dependency of the feeds and the coincidence that two feeds contain the same IoCs without being dependent, the overlap between the feeds is calculated for two different periods. First, the overlap between the feeds regardless of the time of sighting is calculated. This will show the overlap, but doesn't necessarily show the dependency of the feeds. To give a better view on the dependency, the overlap between the feeds is calculated where the time of the sightings is included. We will only count two IoCs as overlapping, if the difference of the first sighting is smaller than one day. We chose one day, because all intelligence feeds used in our research at least update their feed once a day. Thus, if two intelligence feeds have a common source, IoCs from this source should be added within a day to both feeds. Not all intelligence feeds provide an indication of time that tells us when an IoC was firstly sighted. Therefore, the day we started collecting IoCs, a large number of IoCs were added to the dataset of which we didn't know the first date of sighting. The possibility exists that some of these IoCs might have been sighted years ago. For IoCs provided by a feed later on, we could check if it was already in the dataset. If it was not, we saved a timestamp of first sighting ourselves. To exclude the initial IoCs with unknown timestamps, only the IoCs taken after the first day of collecting are used for the overlap matrices.

2.3 Decay Time

An IoC may become less valid over time when it hasn't been seen for a while. To account for this in our scoring model, the score for an IoC of a specific feed is based on a decay time. The decay time uses an exponentially decreasing function, as shown in Equation 2 and in Figure 1. The equation was introduced by Iklody et. al [3][6] to define the decay rate of IoCs. Function $f(x)$ has two parameters τ and δ. τ determines at which x value $f(x)$ is equal to 0. δ represents the decay rate. Because we always want to keep the outcome of the function between zero and one, we take the maximum with zero. Figure 1 illustrates how this parameter determines the change in the slope of the function. If δ has a value between zero and one, the function starts decreasing slowly, but it decreases fast at the end. For a δ above one, it is the opposite, it starts by decreasing fast and decreases slowly at the end. Figure 1 also illustrates that $f(\tau) = 0$.

$$f(x) = max(0,\ 1 - (\frac{x}{\tau})^{\frac{1}{\delta}}) \tag{2}$$

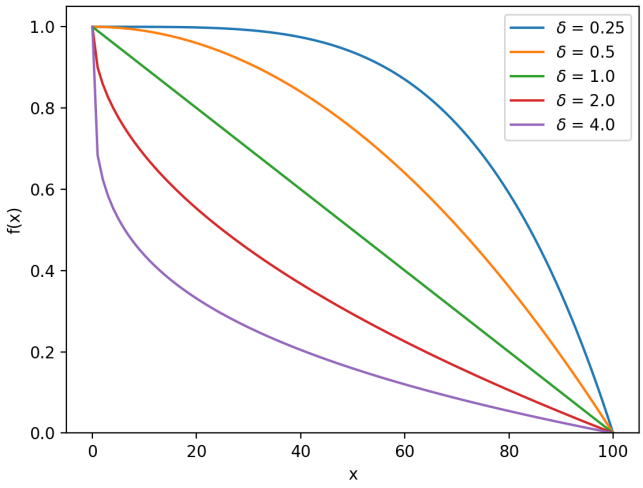

Figure 1: Decay function from Equation 2 with different δ parameter values and a fixed τ value of 100.

To use Equation 1 for decaying the score of an IoC over time, the x value is equal to $T_{current} - T_{lastseen}$, where $T_{current}$ is the current timestamp when the score of the IoC is determined and $T_{lastseen}$ is the timestamp of the last sighting of the IoC. τ represents the time it takes for an IoC to become invalid after its last sighting. The unit of $T_{current}$, $T_{lastseen}$ and τ must be the same. Equation 3 shows the function that is used to make the score of an IoC decay over time. In our research, we choose τ to be seven days, taken the example that an infected machine can be fixed within one week and will be null routed if the machine is not fixed within one week [3]. Furthermore, we choose δ to be 0.5. This means that the score of an IoC will remain almost the same for around two days. After two days, the decaying will start, resulting in a score of 0 after seven days.

$$f(T_{current}, T_{lastseen}) = max(0,\ 1 - (\frac{T_{current} - T_{lastseen}}{\tau})^{\frac{1}{\delta}}) \tag{3}$$

2.4 Source Confidence

Not all intelligence feeds deliver the same quality. We don't know how much false positives each intelligence feed contains in advance, which makes it hard to predict how much confidence we can put into an intelligence feed. T. Schaberreiter et al. defined some some characteristics regarding the quality of the intelligence feeds that can help us define a source confidence (SC) [7]. By using a source confidence we can give higher weights to intelligence feeds with better quality. These characteristics are *Extensiveness*, *Timeliness* and *Completeness*.

The **Extensiveness** (SC_E) defines how much context is given to support the IoCs of an intelligence feed [7]. In Section 2.1 we mentioned that not all intelligence feeds provide timestamps. Therefore we store our own timestamps of the last time we have seen an IoC in a feed. This timestamp is used with the decay time, if an intelligence feed does not provide its own timestamps. Thereby, we give an advantage to feeds that don't provide timestamps, because these timestamps will always be more recent. This also holds for feeds that don't provide their own confidence in an IoC. These feeds also have an advantage in our model, which we will go further into in Section 2.6. With extensiveness, we assign a better source confidence to intelligence feeds that give more context per IoC. Thereby we counteract the aforementioned advantages of feeds with less context. Furthermore, if more context is given, this indicates that a broader analysis has been done on an IoC. Thus, counteracting the advantages of feeds with less context, is not the only purpose of the extensiveness. Properties of intelligence feeds related to the context of IoCs that we will take into account in the extensiveness calculation are:

- Timestamps of last sightings of the IoCs.
- Number of sightings per IoC.
- Description of threats related to the IoCs.
- A confidence score for the IoCs provided by the intelligence feed itself.

Note that only the properties given by an intelligence feed can contribute to the extensiveness. The timestamps provided by ourselves are not taken into account in the extensiveness calculation. Equation 4 shows the function of the extensiveness, where z_s is the total number of IoCs provided by the intelligence feed s, $(o_i)_s$ is the number of contextual properties provided for IoC i and n is maximum number of contextual properties [7]. In our case, $n = 4$.

$$SC_E(s) = \frac{1}{z_s} \sum_{i=0}^{z_s} \frac{(o_i)_s}{n} \tag{4}$$

The **Timeliness** (SC_T) defines how fast an intelligence feed shares its IoCs compared to other intelligence feeds [7]. If a certain intelligence feed shares the same IoCs later than others, the IoCs could be outdated. If a certain intelligence feed is often slow in sharing its IoCs, we will assign less source confidence to it. Equation 5 shows the function of the timeliness, where z_s is the total number of IoCs provided by intelligence feed s, $min(t_i)$ is the timestamp at which the fastest intelligence feed sighted IoC i and $(t_s)_i$ is the timestamp at which s sighted IoC i [7].

$$SC_T(s) = \frac{1}{z_s} \sum_{i=0}^{z_s} \frac{min(t_i)}{(t_s)_i} \tag{5}$$

A problem that will occur is that IoCs first seen a year ago can have influence on the timeliness when compared to current IoCs. An IP address, for example, could have been

relevant a year ago, while the same IP address can be sighted in the present, related to a new threat. To exclude this case while calculating the timeliness, only IoCs sighted within λ days from each other are taken into account. Thus $min(t_i)$ must be within the $[(t_s)_i - \lambda, (t_s)_i + \lambda]$ time interval. In the research of T. Schaberreiter et al. is suggested that the timestamps in equation 5 are expressed in UNIX Epoch Time [7]. The problem with this is that if the difference between timestamps is a few days, there is little to no impact on the timeliness. This is because the timestamps are represented by large numbers, this is demonstrated in Equation 6.

$$\frac{\text{2020-02-02 12:00:00}}{\text{2020-02-03 12:00:00}} = \frac{1580641200}{1580727600} = 0.9999453416 \tag{6}$$

Therefore we suggest to use the timestamp of $(t_s)_i - \lambda$ as the beginning of time instead of the first of January 1970, because only IoCs in the $[(t_s)_i - \lambda, (t_s)_i + \lambda]$ time interval are taken into account. Note that in this case the beginning of time is different per IoC. Equation 7 shows the formula for the new timeliness. Because we choose τ to be seven days for the decay time, as described in 2.3, λ is also equal to seven days. Equation 8 demonstrates that a one day difference, now has much more effect on the timeliness of a feed.

$$SC_T(s) = \frac{1}{z_s} \sum_{i=0}^{z_s} \frac{min(t_i) - ((t_s)_i - \lambda)}{(t_s)_i - ((t_s)_i - \lambda)} = \frac{1}{z_s} \sum_{i=0}^{z_s} \frac{min(t_i) - (t_s)_i + \lambda}{\lambda} \tag{7}$$

$$\frac{(\text{2020-02-02 12:00:00}) - (\text{2020-02-03 12:00:00}) + 7\ days}{7\ days} = \frac{1580641200 - 1580727600 + 604800}{604800} = 0.8571428571 \tag{8}$$

The **Completeness** (SC_C) defines how much an intelligence feed contributes to the total collection of IoCs. Here, the total collection of IoCs is all distinct IoCs provided by the used intelligence feeds together. The number of IoCs in an intelligence feed does not tell us how much false positives an intelligence feed produces. However, if an intelligence feed covers a big part of the total IoCs it indicates that the feed is very useful on its own [7]. Note that this characteristic focuses more on the quantity of an intelligence feed instead of the quality of an intelligence feed. Therefore, trustworthy small scale intelligence feeds could be disadvantaged by this characteristic that might produce high quality IoCs. Equation 9 shows the function of completeness, where z_s is the total number of IoCs provided by intelligence feed s and z_{total} is the total number of distinct IoCs of all intelligence feeds together.

$$SC_C(s) = \frac{z_s}{z_{total}} \tag{9}$$

A characteristic that has not been looked into in previous research [7] is the overlap between an intelligence feed and a whitelist. If an IoC from an intelligence feed also exists in a trusted whitelist, the IoC in the feed is considered a false positive, and the confidence of the intelligence feed will be lower. The **Whitelist Overlap Score** (SC_W) defines how much of the IoCs in an intelligence feed also exist in a trusted whitelist. To calculate the SC_W, the function in Equation 10 is used, which is a variation on Equation 2. In this equation, u_s describes the number of IoCs that exist both in the intelligence feed and the whitelist and z_s represents the total number of IoCs in the intelligence feed. We have also added the parameter ρ, it indicates at which percentage of overlap between the intelligence feed and the whitelist SC_W should be 0. In our research, we don't think an intelligence feed is trustworthy if 10% of its IoCs are whitelisted. Therefore, we use $\rho = 0.1$. To determine

how fast the SC_W decays, δ is used in the equation. The value of δ is used to determine the slope of SC_W based on the size of the overlap between the intelligence feed and the whitelist, as can be seen in Figure 1. It is possible that a host with a whitelisted IP address gets compromised. Therefore, a small overlap between an intelligence feed and a whitelist shouldn't have a large influence on the whitelist overlap score. For that reason, we use a δ value of 0.5.

$$SC_W(s) = max(0,\ 1 - (\frac{u_s}{z_s \cdot \rho})^{\frac{1}{\delta}}) \tag{10}$$

IPs of trusted services change a lot, e.g. cloud services. Therefore it is hard to maintain an IP whitelist. However, whitelists of domains do exist, e.g. the Cisco Umbrella Popularity list [12][1]. We will use the Cisco Umbrella Popularity list to determine the whitelist overlap score. It is important to refresh the IP address translations of the domains before we calculate the whitelist overlap score, because the IP addresses can change quickly. A better way to calculate the overlap of the intelligence feeds with the whitelist would be to make it time dependent. We could keep a history of IP address translations per day. This would allow us to calculate the overlap of the IP addresses in the intelligence feeds sighted on a day in the past, with the whitelisted IP addresses of that day. Because we have not built up this history, the whitelist overlap score is calculated regardless of the time. Improving this will be future work.

2.4.1 Calculating the Source Confidence

To determine the source confidence of the gathered intelligence feeds, a weighted mean is calculated of the four characteristics, as shown in shown in Equation 11. For every characteristic, a weight W_x with a value between 0 and 1 is given based on the importance of the characteristic. First we will give each feature an equal weight of 1. Because we think that completeness is not a useful feature, we also calculate the source confidence where the weight for the completeness is set to 0 and the other weights are set to 1, to see how this influences the source confidence. Finally we calculate the source confidence with a different weight for each feature. We give the whitelist overlap score a weight of 1, because this feature actually focuses on false positives as a indication of the quality of the feed. We give extensiveness a weight of 0.8. This feature does not directly focus on the false positives. However, this feature does indicate how much analysis has been done on the provided IoCs. Provided data by an intelligence feed is important for the scoring model. For example, a provided timestamp of the last sighting is important for the decay time. Therefore, we think that extensiveness is more important than timeliness, which we give a lower weight of 0.6. Finally, we keep the weight of the completeness to 0. These weights will also be used for further experiments in our research.

$$SC(s) = \frac{W_E \cdot SC_E(s) + W_T \cdot SC_T(s) + W_C \cdot SC_C(s) + W_W \cdot SC_W(s)}{W_E + W_T + W_C + W_W} \tag{11}$$

2.5 Calculating the Final Score

To determine the score for an IoC, the IoC is looked up in all the intelligence feeds. For every feed that contains the IoC, a separate score is calculated. This score is based on a possible score (source_score) provided by the feed itself and it decays over time. If the intelligence feed does not provide its own score for the IoC, the source score is set to 1, thus assuming that the intelligence feed is completely confident in its IoCs. This can result in a higher score for an IoC in an intelligence feed that doesn't contain a source score. To prevent giving a higher score for an IoC that provides less information, the source confidence of an intelligence feed is lowered by the extensiveness component if no source score is provided.

Equation 12 shows the calculation of the score of an IoC for a single feed, using the source score and the decay function.

$$score_i = source\_score_i \cdot max(0,\ 1 - (\frac{T_{current} - T_{lastseen}}{\tau})^{\frac{1}{\delta}} \tag{12}$$

After the scores for all separate intelligence feeds are calculated, the final score for an IoC can be determined. The final score is a combination of the scores for all separate intelligence feeds and the confidence scores of these feeds. Equation 13 shows the calculation of the final score, where for all N intelligence feeds that contain the IoC, the score of the IoC is influenced by the source confidence. The final score is the mean of all N intelligence feed scores.

$$final\_score = \frac{1}{N} \sum_{i=0}^{N} source\_confidence_i \cdot score_i \tag{13}$$

The problem with Equation 13 is that every feed has the same amount of influence on the final score. Instead of every feed having the same amount of influence, we want feeds with a higher source confidence to have more influence than feeds with a lower source confidence. To achieve this, the final score needs to be a weighted mean, where the source confidence works as a weight on the final score per feed. The weighted mean of the score is shown in figure 14.

$$final\_score = \frac{\sum_{i=0}^{N} source\_confidence_i \cdot score_i}{\sum_{i=0}^{N} source\_confidence_i} \tag{14}$$

Note that if the IoC only exists in one intelligence feed ($N = 1$), the source confidence would have no influence on the IoC score. To combine the advantage of using the source confidence as influence on the final score, as well as using the source confidence as a weight on the final score, Equation 13 and 14 are combined into Equation 15.

$$final\_score = \frac{\sum_{i=0}^{N} source\_confidence_i^2 \cdot score_i}{\sum_{i=0}^{N} source\_confidence_i} \tag{15}$$

2.6 The Scoring Model

Now that we have explained all the components of our model, we can put it all together. Figure 2 shows a flowchart of our scoring model. This scoring model determines the score of an IoC between 0 and 1, where 0 indicates the highest chance of a false positive and 1 the lowest chance. To determine the score of an IoC, the IoC is first looked up in one or more whitelists. If the IoC exists in a whitelist, the score of the IoC will be 0. Because this research focuses on the quality of the intelligence feeds and not on the whitelists, we assume that the whitelist is trusted and doesn't contain any false positives. It is always a possibility that a whitelisted IP address is reassigned to a malicious owner or the trusted service gets compromised. Including a mitigation for this in our model will be future work. If the IoC does not exists in a whitelist, the IoC is looked up in all the intelligence feeds. For every feed that contains the IoC, a separate score is calculated. After the scores for all separate intelligence feeds are calculated, the final score can be determined using Equation 15.

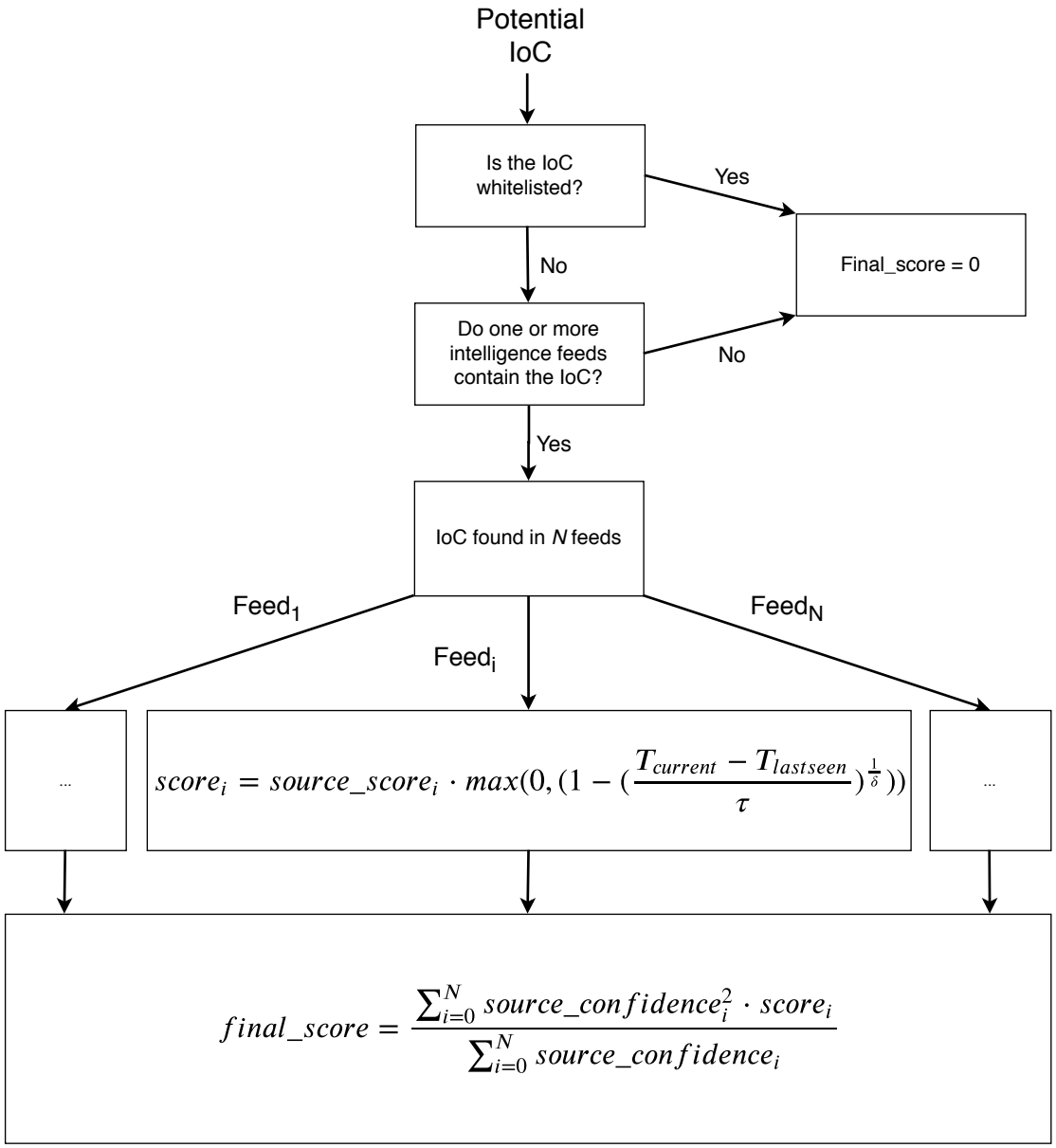

Figure 2: Flowchart of the scoring model

2.7 Putting the Scoring Model in Practice

To put the scoring model in practice, a set of IP addresses is taken from the logs from our web server. These logs contain IP addresses of brute force requests to find some commonly used PHP scripts. Because the web server doesn't use PHP, these IP addresses are likely indicating malicious behaviour and therefore get labeled as IoCs. The scoring model can be used to verify if these IoCs are false positives or not. In this experiment we use the found IP addresses to find out how the scoring model works in practice and notice any shortcomings. From the set of IP addresses, four of those are used to test the scoring model. After the two weeks of gathering IoCs as described in Section 2.1, the scores for the four IP addresses are calculated every hour for a time period of one week. During this week, the data gathered from the intelligence feeds is refreshed once every day, at the same time of the day. This means that the source confidence for each intelligence feed can change once a day, and also the timestamps of the IoCs.

3 Results

First of all, the overlap of the intelligence feeds are shown in Section 3.1 to determine the independence of the intelligence feeds. Furthermore, the extensiveness, timeliness, completeness, and whitelist overlap score are presented in Section 3.2 with the corresponding source confidence for different weights of the characteristics. Finally the behaviour of the scoring model can be seen in Section 3.3 where the IoC Score is calculated over a period of several days.

3.1 The Overlap and Independence of the Intelligence Feeds

Figure 3 shows the overlap matrix of the IP addresses in the intelligence feeds. The matrix shows that there is a high overlap between the Cyber Cure feed ($Feed_A$) and the AbuseIPDB feed ($Feed_B$). Vice versa, the overlap between the AbuseIPDB feed ($Feed_A$) and the Cyber Cure feed ($Feed_B$) is less. This is caused by the difference in total number of IoCs that both feeds have, which is shown in Table 1. Furthermore, the Binary Defense feed has a small overlap with AbuseIPDB and Cyber Cure and vice versa. Note that C&C Tracker doesn't have any overlap with the other intelligence feeds. Also note that this matrix does not show the **dependency** of the intelligence feeds, but only the overlap between the feeds.

Figure 4 shows the overlap matrix of the IP addresses in the intelligence feeds where the difference in time of sighting is smaller than one day, where it is more likely that an overlap shows a dependency between two feeds. The matrix shows that there is an overlap between AbuseIPDB and Cybercure. The other intelligence feeds have an overlap equal or close to zero percent.

	AbuseIPDB	**Binary Defense**	**C&C Tracker**	**Cyber Cure**	**Total Distinct**
Number of IoCs	17283	6677	924	1400	25019

Table 1: The number of IoCs we collected in two weeks time for each intelligence feed. The last column shows the total number of distinct IoCs.

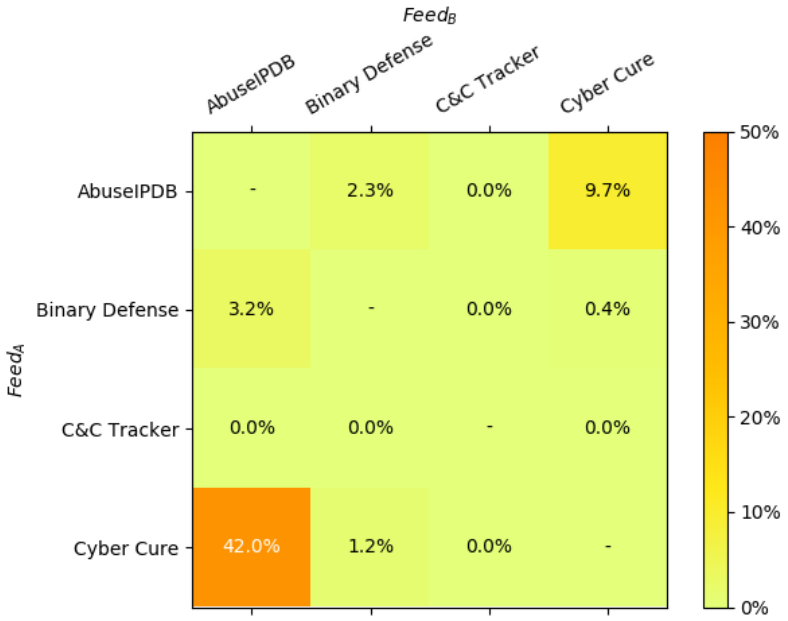

Figure 3: The overlap of IP addresses in the intelligence feeds. The percentages were calculated as in Equation 1.

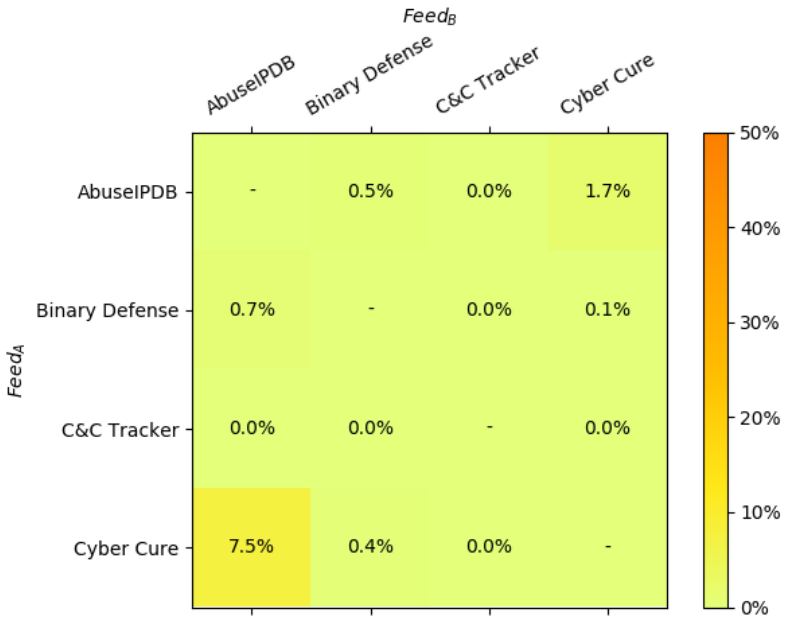

Figure 4: Overlap of IP addresses in the intelligence feeds where the difference in time of the first sighting is smaller than one day. The percentages were calculated as in Equation 1.

3.2 Calculated Characteristics for the Source Confidence of the Intelligence Feeds

Table 2 shows the calculated characteristics and the final source confidence for every intelligence feed. The table shows that Binary Defense and Cyber Cure have an extensiveness of 0.00. Furthermore, both Binary Defense and C&C tracker have a timeliness of 1.00, while AbuseIPDB and Cyber Cure have slightly lower timeless score than 1.00. AbuseIPDB has a much higher completeness than the other feeds. Finally, the table shows that every feed except C&C Tracker has a whitelist overlap score of 1.00. Figure 5 shows the overlap between the used whitelist and the intelligence feeds. C&C Tracker has the most overlap with the whitelist. Also AbuseIPDB and Binary Defense have little overlap with the whitelist. However, due to the small percentage of overlap and Equation 10, the whitelist overlap scores of these two feeds are still 1.00, as shown in table 2. Table 2 also shows the source confidence where the weight of the completeness characteristic is set to 0. For all the intelligence feeds, the source confidence is higher than the source confidence where W_C is set to 1. Also note that there is a larger difference between the source confidence with W_C set to 0 and 1 for C&C Tracker and Cyber Cure than for the other two feeds.

	AbuseIPDB	**Binary Defense**	**C&C Tracker**	**Cyber Cure**
Extensiveness	0.73	0.00	0.50	0.00
Timeliness	0.98	1.00	1.00	0.94
Completeness	0.69	0.27	0.04	0.05
Whitelist Overlap	1.00	1.00	0.61	1.00
Source Confidence	0.85	0.57	0.54	0.50
Source Confidence ($W_C = 0$)	0.91	0.67	0.70	0.65
Source Confidence ($W_E = 0.8$, $W_C = 0$, $W_T = 0.6$, $W_W = 1$)	0.91	0.67	0.67	0.65

Table 2: The calculated characteristics to determine the source confidence and the source confidence itself, for each intelligence feed, as described in section 2.4.

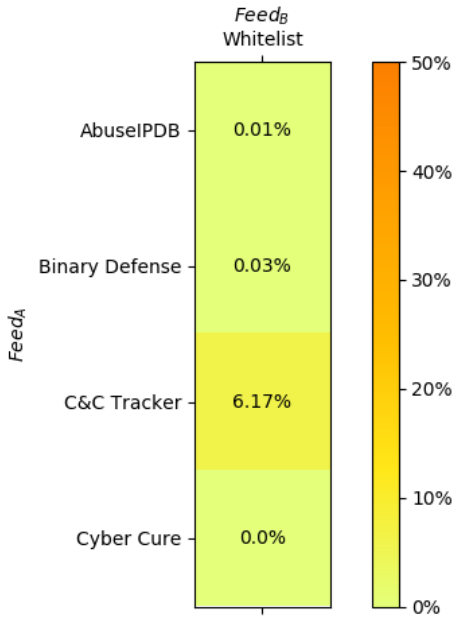

Figure 5: The overlap of IP addresses in the intelligence feeds with the whitelist. The percentages were calculated as in Equation 1.

3.3 Putting the Scoring Model in Practice

Figure 6 shows the scores of four IP addresses taken from the logs of our web server. First, the IP addresses were found in one or more of the used intelligence feeds. Then the figure shows that the score slightly decays over time, due to the decay function. Note that the score from 60.191.66.222 repeatedly goes back to the initial value after a slight decrease. This indicates that the IP address was sighted again in one of the intelligence feeds. IP address 5.101.0.209 was found in two intelligence feeds and then the score decayed. After 40 hours, the score of 5.101.0.209 stays around 0.3, where the IP address was repeatedly sighted in one of the intelligence feeds. Another interesting observation is that the score of 51.75.160.7 increases after 46 hours, because the IP address was found in another intelligence feed. Afterwards, the score decreases. Finally, IP address 5.96.237.174 started with a high score, but decreases over time, eventually reaching a score of 0. An overview of all important events that explain the behaviour of the IoCs can be seen in table 3.

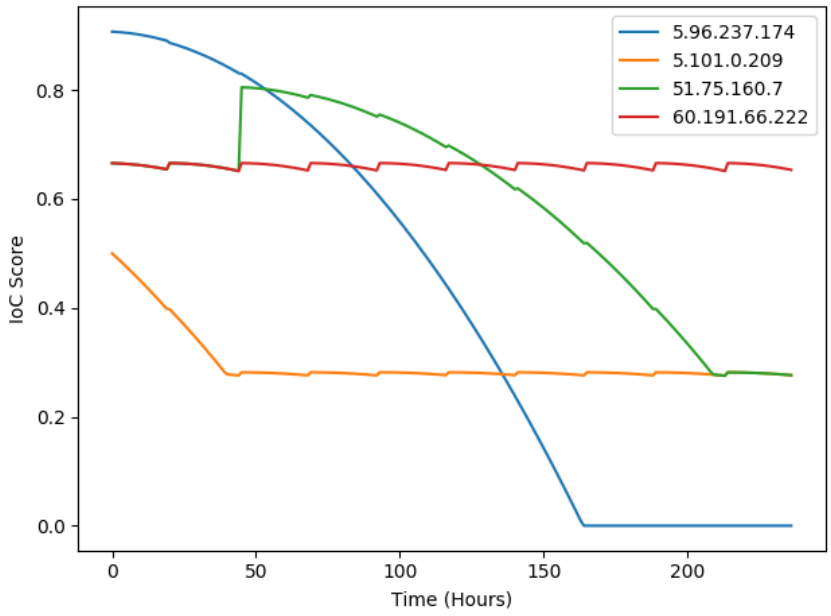

Figure 6: IoC scores that are calculated every hour using the scoring model.

Time	IoC	Event
0	5.96.237.174	Found in AbuseIPDB.
0	5.101.0.209	Found in AbuseIPDB and Binary Defense.
0	51.75.160.7	Found in Binary Defense.
0	60.191.66.222	Found in Binary Defense.
40	5.101.0.209	AbuseIPDB score reaches 0 due to decay. However, Binary Defense keeps the score of the IoC above 0.
46	51.75.160.7	Found in AbuseIPDB.
164	5.96.237.174	AbuseIPDB score reaches 0 due to decay.
210	51.75.160.7	AbuseIPDB score reaches 0 due to decay. However, Binary Defense keeps the score of the IoC above 0.

Table 3: Important events that describe the behavior of the IoC scores in Figure 6.

4 Discussion

4.1 Overlap and Independence of the Intelligence Feeds

In the documentation of the intelligence feeds (Appendix A), we have seen that no pair of intelligence feeds explicitly state that they use the same sources to gather their IoCs. This does not mean that we can conclude they are independent, e.g. AbuseIPDB has anonymous Internet users as a source, which makes it impossible to be completely independent from other intelligence feeds. Therefore we also looked into the overlap between the intelligence feeds. As shown in the overlap matrices in section 3.1, figure 3 shows that there is much overlap between AbuseIPDB and Cyber Cure. The other intelligence feeds barely overlap, which is why we will assume that these feeds are independent. For AbuseIPDB and Cyber Cure, there is a significant difference between the overlap of IoCs where the difference in time of the first sighting is taken into account, as shown in figure 4. This means that the major part of the overlapping IoCs between AbuseIPDB and Cyber Cure are likely independent.

Overlapping feeds are important for the scoring model to determine a score based on multiple feeds instead of only a single feed. Therefore, AbuseIPDB and Cyber Cure would be a good set of feeds to use for the scoring model. For the other feeds, there is almost no overlap, resulting in the scoring model determining most scores based on only a single feed.

4.2 Source Confidence of the Intelligence Feeds

Table 2 has shown us the resulting source confidence and its characteristics for the intelligence feeds. We will discuss the values of the characteristics for the intelligence feed, beginning with **extensiveness**. Binary Defense and Cyber Cure provided only IP addresses to us without more context. Therefore, their extensiveness is zero. AbuseIPDB provided us with a source score, number of reports and a timestamp of the last sighting. However, the last sighting was not given for each IoC. Therefore, the extensiveness of AbuseIPDB is 0.73, instead of 0.75.

When an intelligence feed has little overlap with other feeds, its **timeliness** will be close to one. This is demonstrated by the timeliness of Binary Defense and CC Tracker. AbuseIPDB and Cyber Cure had more overlap, therefore the influence of the timeliness is bigger. AbuseIPDB's timeliness is higher than Cyber Cure's, which means that AbuseIPDB was on average faster with sharing new IoCs than Cyber Cure.

In Section 2.4 we mentioned that the **completeness** focuses on the quantity and not on the quality of IoCs. In Table 2 we have seen that the source confidence is lowered for each intelligence feed when completeness is used. The completeness of a feed is never one, because none of the feeds contain all IoCs. This would also be an unrealistic expectation of an intelligence feed. Table 2 also showed us the source confidence, with and without the completeness taken into account. Comparing the two different source confidences, we see that for each of the intelligence feeds, the source confidence is lowered based on quantity when completeness is used. If a big intelligence feed with mostly false positives would be used, with low extensiveness, timeliness and whitelist overlap score, the completeness could even have a positive effect on this feed, which is undesirable. For these reasons, we rate completeness as a bad characteristic for the source confidence and gave the completeness a weight of 0 for the rest of the experiments.

The last characteristic of the source confidence is the **whitelist overlap score**. Figure 5 has shown that AbuseIPDB, Binary Defense and Cyber Cure have little to no overlap with the IP address whitelist. Because we used a low decay rate ($\delta = 0.5$), the little overlap has almost no influence on the whitelist overlap score. Therefore the whitelist overlap scores of AbuseIPDB, Binary Defense and Cyber Cure are 1.00. However, C&C Tracker overlaps for 6.17% with the whitelist. This indicates that even more IoCs of C&C Tracker could possibly be false positives. Therefore, with a whitelist overlap score of 0.61 we will put less trust in this intelligence feed than in AbuseIPDB, Binary Defense and Cyber Cure. This is caused by using $\rho = 0.1$, which makes the whitelist overlap score already zero, when 10% of an intelligence feed overlaps with the whitelist.

4.3 Putting the Scoring Model in Practice

As shown in Figure 6 and Table 3 in Section 3.3, the score for a set of IP addresses are calculated over a period of time. The IoC 5.96.237.174 was only found in AbuseIPDB. This feed has a 0.91 source confidence. Therefore the score of 5.96.237.174 starts high. This intelligence feed provides its own timestamp of when an IoC was last sighted. Therefore the decay of its score over time is working well. However if we look at 60.191.66.222, this IoC was only found in Binary Defense Banlist. This feed has a source confidence of only 0.67,

which is why the score of 60.191.66.222 starts much lower than 5.96.237.174. Binary Defense Banlist does not provide its own timestamp. Therefore we produced our own timestamp of last sighting, which we updated every time the IoC was found in Binary Defense Banlist. As Binary Defense Banlist has never removed an IoC in the three weeks we have accessed it, we assume that it only adds new IoCs and does not maintain old IoCs that may have lost their malicious users. This results in a loop where an IoC starts with a $score_A$, decays for a day, after which the score resets to $score_A$, and starts to decay from the same score again. We will call this kind of behavior *decay loops*. The behavior of 60.191.66.222 in Figure 6 is an example of a decay loop. 51.75.160.7 began with the same problem as 60.191.66.222. However, after 46 hours it was also found in AbuseIPDB, resulting in a higher score at first. Afterwards, the score of AbuseIPDB will decay to zero, while the score of Binary Defense stays the same. This results in the same situation as 5.101.0.209, where the score again ends up in a decay loop. However, between 0.282 and 0.276 this time. Thus, by using the source confidence as a weight for the final score, we have the advantage that the final score decays when a trustworthy feed says the IoC is outdated.

To prevent the behaviour of decay loops, intelligence feeds should provide their own timestamps of last sightings of an IoC, or they should remove their IoCs when these don't show malicious behaviour anymore. Another solution would be to use the timestamp of the first sighting, when an intelligence feed does not maintain old IoCs. The disadvantage is that an IoC of this feed will only be useful for a τ time period, and never again. The advantage of this is that we are sure that we don't base any scores on outdated IoCs.

4.3.1 Shortcoming of the Current Scoring Model

A final remark on Equation 15 to calculate the final score is that we don't distinguish two important reasons why an IoC score can vary for different intelligence feeds. The first reason is that a score can be low, because the source confidence of the associated intelligence feed is low. The second reason is that a score can be low due to decay, because the associated feed has not sighted the IoC for some time. The first case is not handled correctly in our model. Take 51.75.160.7 in Figure 6 as an example. After 46 hours, the IoC is present in both Binary Defense Banlist and AbuseIPDB. Both intelligence feeds are used in the weighted mean calculation to calculate the final score. Because of the weighted mean, if Binary Defense Banlist did not contain this IoC, the final score would be higher. However, if both AbuseIPDB and Binary Defense Banlist contain an IoC, it is more likely that an IoC is a true positive. A better model would give a higher score than when the IoC is only present in AbuseIPDB. However, the second case is handled correctly, because of the weighted mean. Now take 5.101.0.209 in Figure 6 as example. If this IoC was only present in Binary Defense Banlist, then the score of this IoC would behave the same as that of 60.191.66.222. However, because AbuseIPDB with higher source confidence says that this IoC is outdated, the final score will be low.

A possible solution to counteract the incorrect handling of the first case is to generate two final scores for an IoC instead of one. The first final score is the score that is based on the source confidence. With this score, it is possible to decide whether an IP address is/was an IoC. Furthermore, this score based on the source confidence should be calculated in such a way that if multiple intelligence feeds contain an IoC, this will have a positive effect on the score. The other final score is based on the decay time. With this score, it is possible to decide how old an IoC is. Together, both final scores can be used to decide if an IP address ever was an IoC, and if it still is.

5 Conclusion

The aim of this research is to combine open threat intelligence feeds to design a scoring model for IoCs to reduce the number of false positives. The first step in the process of designing the scoring model is finding independent intelligence feeds. The results show that we can assume that the used feeds are independent, but didn't have much overlap. Ideally, we want more overlapping independent intelligence feeds.

The scoring model is time dependent due to the decay time function, taken from previous research [3][6]. We saw that time dependency in the scoring model didn't work when the intelligence feed doesn't provide a timestamp of the last sighting of the IoC or doesn't remove an IoC when it is not relevant anymore, resulting in decay loops.

To decide if we can trust an intelligence feed, we calculated a source confidence that is based on features like the extensiveness, timeliness, completeness, and whitelist overlap score. Experimenting with these characteristics shows that a characteristic like completeness is a bad characteristic to determine the quality of an intelligence feed, because this characteristic is based on the quantity of the intelligence feed instead of the quality. The other characteristics are useful, but their weights to calculate the source confidence need further research for optimization.

To calculate a final score for an IoC based on the score from multiple intelligence feeds, the source confidence influences the final score. Also, the source confidence is used to calculate a weighted mean for the final score, where intelligence feeds with a higher confidence have more influence on the final score.

These challenges have brought us to the current design of the scoring model. With the current function for the final score, the source confidence still influences the score of an IoC if it is only found in one intelligence feed. Furthermore, when an IoC is found in multiple intelligence feeds, the final score is influenced the most by the feed with the best source confidence. These are some good advantages. However, the scoring model still has some shortcomings. The model should prevent scores from being stuck in decay loops. Furthermore, the current scoring model provides one final outcome, but it would be better to split it up into two scores, showing the score based on the source confidence and on the age of an IoC.

5.1 Future Work

The results show that most of the intelligence feeds that are used in this research don't have much overlapping IoCs, resulting in the scoring model determining a score based on only a single feed. For future research, independent feeds with more overlap can be used in combination with the scoring model. Furthermore, in our research the dependency of the intelligence feeds is only based on the overlap within a period of time. For future work the dependency of feeds can be researched more in depth.

Also, in our research the confidence of the intelligence feed is based on the extensiveness, timeliness, completeness, and the whitelist overlap score. In future work, different characteristics could be used to determine the quality of the intelligence feed. Many formulas presented in our research are dependent on certain parameters, like the parameters for the decay time (δ and τ) and the weights for the characteristics to calculate the source confidence. In future work, these parameters could be researched more in depth to find the optimal value for each parameter, resulting in a lower number of false positives when used in combination with our scoring model.

IP addresses are not the only types of IoCs, in the future other types can be researched in combination with this model, or an improved model. Hashes of malware files and domain names are other interesting IoC types.

Finally, if an IoC appears in a whitelist, our scoring model gives the IoC a score of 0 regardless of other intelligence feeds. For future work, the confidence of whitelists could be researched more in depth like the intelligence feeds in this research. This will result in a more sophisticated scoring model where an IoC that exists in a whitelist doesn't necessarily gets a score of 0. This could be useful because IP addresses in whitelists could be compromised, resulting in a whitelist containing a malicious IP address.

References

[1] hslatman, "awesome-threat-intelligence." `https://github.com/hslatman/awesome-threat-intelligence`, 2019.

[2] X. Liao, K. Yuan, X. Wang, Z. Li, L. Xing, and R. Beyah, "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*, pp. 755–766, ACM, 2016.

[3] A. Iklody, G. Wagener, A. Dulaunoy, S. Mokaddem, and C. Wagner, "Decaying indicators of compromise," *arXiv preprint arXiv:1803.11052*, 2018.

[4] C. Wagner, A. Dulaunoy, G. Wagener, and A. Iklody, "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*, pp. 49–56, ACM, 2016.

[5] A. Dulaunoy, "Misp threat sharing platform (sha2017)." `https://youtu.be/M7JMGOtxOOo?t=1411`, 2017.

[6] S. Mokaddem, G. Wagener, A. Dulaunoy, and A. Iklody, "Taxonomy driven indicator scoring in misp threat intelligence platforms," *arXiv preprint arXiv:1902.03914*, 2019.

[7] T. Schaberreiter, V. Kupfersberger, K. Rantos, A. Spyros, A. Papanikolaou, C. Ilioudis, and G. Quirchmayr, "A quantitative evaluation of trust in the quality of cyber threat intelligence sources," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pp. 1–10, 2019.

[8] AbuseIPDB, "Abuseipdb." `https://www.abuseipdb.com/`, 2020.

[9] B. Defense, "Binary defense banlist." `https://www.binarydefense.com/banlist.txt`, 2020.

[10] B. Consulting, "Cc tracker." `http://osint.bambenekconsulting.com/feeds/c2-ipmasterlist.txt`, 2020.

[11] CyberCure, "Cybercure free intelligence feeds." `http://api.cybercure.ai/feed/get_ips`, 2020.

[12] C. Umbrella, "Cisco umbrella top 1 million." `http://s3-us-west-1.amazonaws.com/umbrella-static/index.html`, 2020.

[13] AbuseIPDB, "Abuseipdb api." `https://docs.abuseipdb.com/#introduction`, 2020.

[14] D. Kennedy, "Artillery 1.4 released – new major features." `https://www.binarydefense.com/artillery-1-4-released/`, 2015.

[15] E. Labs, "Et labs: Emerging threats compromised ips." `http://rules.emergingthreats.net/blockrules/compromised-ips.txt`, 2020.

[16] Malc0de, "Malc0de: Ip blacklist." `http://malc0de.com/bl/IP_Blacklist.txt`, 2020.

[17] Abuse.ch, "Abuse.ch zeus tracker." `https://zeustracker.abuse.ch/blocklist.php?download=badips`, 2015.

[18] Abuse.ch, "Abuse.ch palevo tracker." `https://palevotracker.abuse.ch/blocklists.php?download=ipblocklist`, 2015.

A Description of the Intelligence Feeds

AbuseIPDB is an intelligence feed where everybody can report IP addresses they expect to be malicious. AbuseIPDB uses these reports to give a score to IP addresses based on the number of reports. We can use this score as the *source_score* shown in Figure 2. The service of AbuseIPDB allows us to look up IP addresses and find a *source_score*, a timestamp of the last report and the number of reports [13]. Each day as a free user of AbuseIPDB, a blacklist of 10000 IP addresses can be requested. Initially AbuseIPDB gave us 10000 IP addresses, then each day afterwards, an average of 430 new IP addresses were provided.

Binary Defense Banlist combines four intelligence feeds to one list which is updated every two hours [14]. These two of the four intelligence feeds are a compromised IPs feed from ET Labs [15] and an IP blacklist from Malc0de [16]. The last two feeds are not active anymore, but were both botnet trackers from abuse.ch [17][18]. Binary Defense Banlist only shares the IP addresses, no more context is delivered per IoC. Binary Defense Banlist provided us initially with 2414 IP addresses, then each day afterwards, an average of 305 new IP addresses were given.

C&C Tracker by Bambenek Consulting provides feeds of IP addresses and domain names labeled with different malware families [10]. For every malware family, there is a feed available. There is also a master feed available containing all IP addresses from all specific family feeds. Besides the master feed, there is also a master feed only containing IP addresses from family feeds. In our research, the master feed containing all IP addresses is used. This feed is generated every 60 minutes. Besides the IP addresses, the feed also provides a timestamp of the last sighting of the IoC and a description about the feed of the specific malware family. It is unknown to us how the IoC feeds are generated. C&C Tracker provided us initially with 535 IP addresses, then each day afterwards, an average of 28 new IP addresses were given.

Cyber Cure provides a free API where 100 IP addresses can be requested every day. The list of IP addresses gets updated every several hours. The website of Cyber Cure states that "several indicators that are being collected from the internet and provided by commercial vendors running honeypots and honeynets" [11]. However, the specific sources are not listed. The feed only provides an IP address and no additional information. Each day as a free user of CyberCure, a blacklist of 100 IP addresses can be requested. Initially Cyber Cure gave us 100 IP addresses, then each day afterwards, also 100 new IP addresses were provided.