Enriching DNS Flows with Host-Based Events to Bypass Future Protocol Encryption

Stanislav Špaček[1,2], Daniel Tovarňák[2], and Pavel Čeleda[2]

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Institute of Computer Science, Masaryk University, Brno, Czech Republic

Abstract. Monitoring of host-based events and network flows are the two most common techniques for collecting and analyzing cybersecurity data. However, events and flows are either monitored separately or correlated as alerts in higher aggregated forms. The event-flow correlation on the monitoring level would match related events and flows together and enabled observing both data in near real-time. This approach allows substituting application-level flow information that will not be available due to encryption, which is being employed in a number of communication protocols. In this paper, we performed the event-flow correlation of the DNS protocol. We developed a general model that describes the relation between events and flows to enable an accurate time-based correlation where parameter-based correlation is not feasible. Based on the model, we designed three event-flow correlation methods based on common parameters and times of occurrence. We evaluated the correlation methods using a recent and public dataset, both with and without the extended flow information, to simulate DNS flow encryption. The results of the method combining parameter-based and time-based matching show that matching related DNS events to flows is possible and substitutes the data that might soon be lost in encryption.

Keywords: HIDS · NIDS · Event-flow correlation · DNS · Encrypted network traffic.

1 Introduction

Monitoring of network flows using a Network Intrusion Detection System (NIDS) is a well-described and widely used technique of data analysis for cybersecurity. Over time it has evolved to use targeted Deep Packet Inspection (DPI) to extend the flow information with data from the application layer protocols. However, the recent shift to information privacy moved most of the formerly available data behind end-to-end encryption. If a network flow is encrypted, the enrichment by DPI is severely hampered, and the output of the monitoring is not as information-rich as in the past. As a result, NIDS threat detection and security analysis of the encrypted network flows requires new approaches to support cybersecurity operations.

On the other hand, monitoring of host-based data by a Host-based Intrusion Detection System (HIDS) is another well-known technique to collect and analyze

data in the form of events. The event monitoring provides information about the events taking place directly on the end-point devices. However, the dependency on the end-point devices is the drawback of this method. If attackers take control over the device, they may modify, remove, or otherwise tamper with the events. The flow monitoring may have its own dedicated infrastructure and is more robust to tampering attempts.

Using both NIDS and HIDS simultaneously, an occurrence may be observed from two different vantage points providing different information [3]. We highlight three main benefits of this approach:

- Providing cybersecurity operators with the correlated data through an appropriate presentation layer will help them quickly identify all data related to a specific occurrence. That will improve their situational awareness and thus speed up the cybersecurity incident response process.
- Any tampering with one monitoring source can be detected by correlation with the other source. Detection of tampering attempts may assist in defense against stealthy persistent attackers.
- The approach allows enriching encrypted flows with host-based data transparently for tools deployed on higher layers. Thus an existing anomaly detection or analysis infrastructure may be used to process encrypted traffic.

To examine the possibilities of correlating related events and flows, this paper aims to answer the following research questions. The first question is, *how reliably can the related DNS events and flows be matched?* The second question is, *what otherwise unavailable monitoring data can the event-flow matching provide when the DNS flow is encrypted?* To answer the first question, we developed a model that describes the relation between events and flows. Based on this model, we designed three matching algorithms using common parameters and times of occurrence. Then we evaluated the algorithms on a recent dataset collected during a complex cyberdefense exercise [11]. To answer the second question, we ran the matching algorithms on a dataset including simulated encrypted DNS flows.

Our results show that while parameter-based event-flow matching performs very well with DPI-enriched DNS flows, performance with basic DNS flows is lacking. The time-based matching, although generally less accurate, is not affected by DPI enrichment and thus performs better on basic or encrypted flows. A matching method combining these two approaches produced the most accurate results on both encrypted and unencrypted flows. We provide algorithms used in this research as open-source software to ensure repeatability and verifiability of our results [15].

The rest of the paper is structured as follows. Section 2 sets the context of this paper by presenting the related work. Section 3 introduces the flow and event capture model, that demonstrates the relation between events and flows. Section 4 specifies the event-flow matching methods tested in this paper. Section 5 describes the actions performed on the public dataset to preprocess it before measurements. Section 6 discusses the results of the matching methods when run on the dataset. Section 7 summarizes lessons learned. Section 8 concludes the paper with answers to the posed research questions.

2 Related Work

There are three categories of related work. First, we discuss works that researched information leakage of the encrypted DNS network traffic, as these methods could be used to directly access the extended flow features of encrypted DNS flows. Second, we describe works that already focused on the event-flow correlation in the past. Finally, we mention the Collaborative Intrusion Detection Systems that combine aggregated alerts from both NIDS and HIDS.

Currently, there are two widely used approaches to encrypting DNS traffic – the DNS over TLS and the DNS over HTTPS protocols. None of these protocols allows the DPI enrichment of DNS flows, leaving only the basic flow features available. Subsequently, the performance of the extended flow-based NIDS is limited [13]. Possible weaknesses of the DNS encryption to flow analysis and website fingerprinting were researched in several prior works [7, 9]. Their results show that extended flow feature extraction, e.g., the queried domain name, is generally possible. However, Bushart et al. mentioned techniques that may make this approach more difficult or even unfeasible in the future [2]. The main disadvantage of this approach is taking the role of an attacker against the monitored traffic. If host-based monitoring of the DNS resolver is an option, the extended flow features may be supplemented from server logs. However, a direct link connecting a network flow to events that it caused needs to be identified first.

The direct correlation of network and host-based data had been explored in the past. Dreger et al. provided a framework for enriching network monitoring with host-based context and noted the advantage of overcoming encryption [4]. More recently, Haas et al. proposed the Zeek-osquery platform for correlating network flows with the originating processes and users [5]. Henderson et al. proposed a time-based correlation algorithm and confirmed that this approach is viable by testing it with real network data [6]. They also discussed the limitations of the event-flow correlation. However, they investigated the correlation solely from the malicious event standpoint and did not consider network flow features aside from its start time, end time, and source. Furthermore, previous works considered the captured times of all correlated occurrences as synchronized and accurate, which is usually not the case when correlating data from devices across the network, as described by Brilingaite et al. [1].

Collaborative Intrusion Detection Systems (CIDS) comprise of several cooperating NIDS and HIDS that collect, share, and aggregate monitoring data for complex analysis of the state of the network. Taxonomy of the CIDS had been provided in a survey by Vasilomanolakis et al. [12]. However, the CIDS usually correlate events of higher, already aggregated forms, e.g. attack signature alerts, and are closely specialized in a particular area [10, 14]. To our knowledge, no direct link between a network flow and events that it caused had been proposed yet.

3 Event and Flow Capture Model

This section introduces the event and flow capture model that demonstrates the relation between events and flows captured at different vantage points in the network [3]. Specifically, we define the relations between the times of occurrence of events and flows to allow designing time-based event-flow matching algorithms. Such an algorithm could be based on the start of the flow, end of the flow, and time of occurrence of an event. An event and a flow would be considered related if the event's time of occurrence fell into the interval between the flow start and the flow end. However, the environment where the events and flows are observed is too complex for this simple method to work.

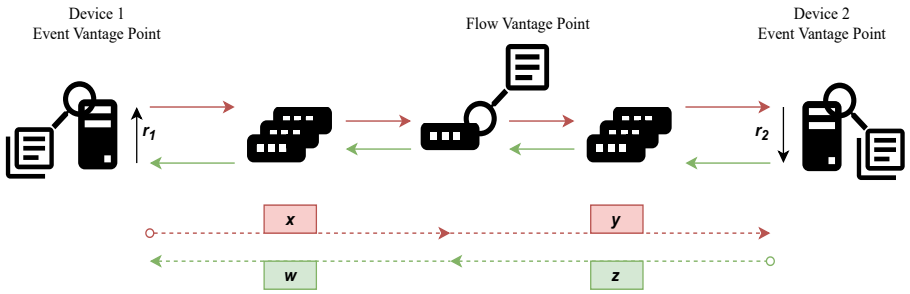

Fig. 1. The general event and flow capture model demonstrates the differences between the time of event occurrence and the time when the associated flow was observed; w, x, y, z are data transfer time intervals of various lengths; r is a time interval needed by a device to process a request and compose a response.

Time is a critical parameter when examining relations between data collected from different sources. However, when the data sources are physically separate, determining the time of occurrence becomes a problem, as time synchronization, network transfer times, and various delays need to be accounted for [1]. We have created a model representing the environment where events and flows are acquired. Furthermore, we have identified several obstacles that stand in the way of simple event-flow matching. The general event and flow capture model is depicted in Figure 1.

The model shows a simple network of two devices communicating over a link consisting of an arbitrary number of hops. Events are generated and monitored on both the communicating devices, and a flow monitoring probe is installed on one of the hops. It is apparent from the model that simple time-based matching is not sufficient, as events and flows are captured at different places and at different times. The issues arising from this can be demonstrated in the sequence of actions during the start (Table 1) of the communication.

Table 1 represents the sequence of event and flow capture during communication start. The communication starts at time t_0 and a correspondign event on

Table 1. Event and flow capture sequence at the start of communication.

Step	Time	Action	Observed Data
1	t_0	request sent	device 1 event
2	$t_0 + x$	in-flow created	flow start
3	$t_0 + x + y$	request received	device 2 event
4	t_1	response sent	device 2 event
5	$t_1 + z$	out-flow created	flow start
6	$t_1 + z + w$	response received	device 1 event

the initiating device (Device 1) is created. However, the flow is captured later, when it enters the flow vantage point at time $t_0 + x$. The event on the responding device (Device 2) is captured even later, at $t_0 + x + y$. It is evident that the time of the flow capture does not mark the start of the flow, but the start increased by an arbitrary network travel time interval. An analogical situation occurs when a communication is terminated. Consequently, an event may be captured before the related flow is created and, analogically, after the related flow is terminated. This issue is mitigated by the use of biflow instead of flow. The biflow considers flows of both directions as parts of one communication and allows matching events from the start of the request flow to the end of the response flow.

Guaranteeing time synchronization between all the monitoring devices is another issue. The NTP protocol is still widely used to provide time synchronization as it can achieve millisecond precision with minimal drift. This precision proved to be sufficient for matching DNS request flows and events in a dataset encompassing circa 200 devices used in this paper. However, in larger environments and with chattier protocols, even a millisecond precision might not be enough to establish a relation between corresponding events and flows.

4 Event and Flow Matching Methods

The event and flow capture model shows that simple time-based event-flow matching does not guarantee good matching accuracy. The event-flow matching methods introduced in this chapter are optimized to pair the DNS request events and flows specifically. The general principle of these methods can be applied to other communication protocols. However, as network communication protocols are too diverse, it is improbable that a single event-flow matching method will work for every protocol.

4.1 Parameter-Based Matching Method

Matching based on common parameters present both in the logged event and in the captured flow is a natural approach to event-flow matching. Both the DNS request flows and the DNS request events contain four common parameters in total. The parameters are the *destination IP address*, *source IP address*, *source port*, and queried domain name (*qname*). This 4-tuple of parameters is

usually sufficient to pair a DNS request event with a related flow. The matching algorithm is described by Algorithm 1.

The common 4-tuple provides a clear relationship between an event and a flow, but it does not always identify an event-flow pair uniquely. A DNS client may reuse a *source port* in any future request, even in a request for the same *qname* as in the past. This results in cross-pairing, where two or more flows share two or more events that cannot be possibly related when considering their times of occurrence. The cross-pairing is especially pronounced with DNS servers queried only for a limited domain pool. The issue is mitigated if a time-window for pairing is specified, and the events that fall outside of the window are not paired. In our dataset, a time window of five minutes proved to be sufficient to suppress most of the cross-pairs.

Algorithm 1: Parameter-Based Matching

```
for each flow ∈ flows do
  for each event ∈ events do
    if flow.dst_ip = event.server_ip & flow.src_ip = event.client_ip &
    flow.src_port = event.client_port & flow.qname = event.qname then
      match flow with event
    end if
  end for
end for
```

The main drawback of the parameter-based matching method is its reliance on a parameter that might not be available. The *qname* is a part of the extended flow information, and it needs to be extracted from the DNS packets by DPI on the flow monitoring probe. If the data cannot be extracted, the *qname* will be unavailable. In our dataset, nearly a fifth of the DNS flows was missing the extended flow information. Any such flows and their related events cannot be paired by this matching method.

The case of the missing *qname* is a sample of the issues caused to network flow monitoring by encrypted communication used on a large scale. If a flow is encrypted, its packets cannot be examined and parsed by DPI for extended flow information. The amount of information available in such a network flow is limited. In these cases, a successful pairing of a flow with its related event(s) provides information on the purpose of the communication that is no more accessible by just flow monitoring.

If the *qname* is missing, the method falls back to pairing based on the three remaining parameters. However, they are not enough to provide a reliable pairing (Section 6). A different and more accurate approach is needed, and a suitable candidate might be refined from this method by extending the use of the time-window specified earlier to filter out cross-pairings.

4.2 Time-Based Matching Method

The time-based matching method copes with the missing *qname* parameter and consequently hampered matching accuracy by introducing a comparison of times of capture for both the events and flows. The idea behind it is simple; if the three remaining common parameters match and the event's time of occurrence falls within the time interval defined by the start and the end of the flow, the event and the flow are considered related. However, such a simple time-based matching method does not consider the time differences, as demonstrated by the event and flow capture model, and needs to be modified.

To compensate for the different vantage points, random packet travel times, and drift in time synchronization between devices, defining a toleration time-window is necessary. This time window has two components – *event earliness* and *event lateness*. The *event earliness* is defined as a time value, indicating how much earlier may an event occur before the start of the request flow to be still considered related to that flow. Analogically, the *event lateness* is defined as a time value, indicating how much later may an event occur after the end of the request flow to be still considered related. The algorithm of time-based matching is described by Algorithm 2.

Algorithm 2: Time-Based Matching

```
for each flow ∈ flows do
  for each event ∈ events do
    if flow.dst_ip = event.server_ip & flow.src_ip = event.client_ip &
    flow.src_port = event.client_port &
    flow.start − event_earliness ≤ event.start &
    flow.end + event_lateness ≥ event.start then
      match flow with event
    end if
  end for
end for
```

The exact values of the *event earliness* and *event lateness* depend on the specific features of the network and devices that generate the data to be paired. They are affected mainly by the packet travel time between the flow vantage point and the DNS server, the time it takes the DNS server to respond to a query, and by the difference in local time of the flow vantage point and the DNS server. The smaller these differences are, the lower may be the *event earliness* and *event lateness*, and the more accurate will be the matching results. With the increasing time-window, the probability of pairing unrelated events and flows also increases. As a result, this matching method needs to be balanced between pairing unrelated events and flows (false positive) or missing a related pair (false negative).

4.3 Combined Matching Method

The combined matching method utilizes both the parameter-based and time-based matching approaches to compensate for their weaknesses. When pairing flows with valid extended flow information, including the *qname*, it uses the parameter-based matching. If it encounters an encrypted flow or a flow with missing or corrupted extended flow information, it falls back to the time-based matching. This way, it maintains the highest possible pairing accuracy for flows with all known parameters while retaining the ability to pair flows with partially known parameters.

The usability for both encrypted and unencrypted network flows and good matching accuracy makes this matching method a suitable candidate for real network traffic. Its performance has been tested, and the results show that it is indeed the preferred pairing method to match related DNS events and flows.

5 DNS Communication Dataset

The dataset used for evaluation in this paper is the Cyber Czech 2018 dataset [11]. It contains both the network flows and event logs, and all the devices in the dataset are time-synchronized with millisecond precision. The data in the dataset was captured during a cybersecurity exercise and generated by the actions of real users.

The initial dataset contained a total of 94 834 DNS biflows and 153 286 DNS events. This amount of DNS data is not final, however. We filtered out obvious white-noise data, as would do any current NIDS or HIDS. Most notably, we filtered out events that were not a DNS request, request events related to flows that circumvented the monitoring probe, and DNS flows to external DNS resolvers. In summary, any data where it was certain that it could not possibly have its either event or flow counterpart in the dataset and thus a correlation attempt would be futile. The filtering algorithm is described in detail in our open-source software [15].

Table 2. The DNS Request Datasets Used for Measurements.

	Events	Extended Biflows	Biflows
all-information	65 682	54 819	0
reduced-information	65 682	27 410	27 409

The resulting smaller DNS request dataset, containing events and flows that can be paired with high probability, is labeled the *all-information dataset*. To examine how the event and flow matching methods perform on basic (non-extended) DNS flows, we created one more dataset. The *reduced-information dataset* contains the same events and flows as the *all-information dataset*. However, half of the flows have been deprived of the extended flow information and

reduced only to the basic flow features as if they were encrypted. The final contents of both datasets are summed up in Table 2.

6 DNS Event-Flow Matching

The performance of the matching methods on both datasets may be inspected in Tables 3 and 4. The measured metrics are *accuracy*, *precision*, *recall*, and *F1-score*. These metrics describe the performance even on a class-imbalanced data set with a high prevalence of true negatives (non-matches), which will be the usual input of the matching methods [8]. We provide supplementary materials; the code and both the filtered datasets used in this research, along with the necessary documentation to verify our results [15].

The metrics are affected by the properties of the dataset. Specifically, by the low number of unpaired events and flows, and by short server response times relatively to longer pauses between separate DNS queries. The achieved *precision*, *recall*, and subsequently *F1-score* values would be lower if the dataset filtering had been more lenient. Moreover, the *accuracy* metrics specifically was affected by the high number of true negatives in the dataset. The count of true negatives reached over 3.6 billions, a value several orders of magnitude higher than the 51 740 true positive pairs.

Due to the strict dataset filtering and therefore little room for error, the results are often very close. However, with the high volume of processed data, even a small difference in a metric might mean a much lower number of false positives or false negatives for the better performing method.

Table 3. Event-Flow Matching Methods' Results on the All-Information Dataset

	Param-Based	Param-Based (no qname)	Time-Based (200 ms)	Combined (200 ms)
Acc	1,000000	0,999998	0,999998	1,000000
Pre	1,000000	0,888118	1,000000	1,000000
Rec	1,000000	1,000000	0,999826	1,000000
F1	1,000000	0,940744	0,999913	1,000000

Table 4. Event-Flow Matching Methods' Results on the Reduced-Information Dataset

	Param-Based	Param-Based (no qname)	Time-Based (200 ms)	Combined (200 ms)
Acc	0,999992	0,999998	0,999998	0,999999
Pre	1,000000	0,888118	1,000000	1,000000
Rec	0,500000	1,000000	0,999826	0,999903
F1	0,666667	0,940744	0,999913	0,999951

6.1 Parameter-Based Matching Method

The parameter-based matching method produces the best results when run on a dataset with valid extended flow information – known *destination IP address*, *source IP address*, *source port*, and *qname*. Because it considers all the parameters common to a DNS request event and flow, it can create a list of event-flow pairs that are related without a doubt. This method had been run on the *all-information dataset*, where it identified 51 740 pairs. This list of pairs is used throughout this paper as a ground truth. The results can be found under the key *Param-Based* in Tables 3 and 4.

Out of the 51 740 pairs, only ten pairs were cross-paired. The cross-pairing is caused by a client reusing a port to query the same server for the same domain in an interval so short that it cannot be distinguished which event is related to which flow. The matching was unique for the remaining pairs, so a flow was paired with only a single event and vice versa. This is caused by DNS requests' specific format, where one flow usually corresponds with one request for a single domain.

While the parameter-based matching method achieved the most accurate results on the *all-information dataset*, it was unable to create a pair if any of the parameters had been missing. Consequently, this method was able to create only 50 % of the pairs when run on the *reduced-information dataset*, producing the worst results of all tested methods. This drawback makes this method unsuitable for use in real traffic, where the collected information might be incomplete.

To lower the reliance on the *qname*, the parameter-based matching method had been modified to match by the three remaining parameters. Then it had been run again on both the datasets and the results can be found under the key *Param-Based (no qname)* in Tables 3 and 4. The *precision* is its weak spot, producing a high number of false positives. This is a serious drawback, as low *precision* may clutter the result with false positives so that important information may get overlooked. On the other hand, it is not influenced by flow encryption.

The parameter-based event-flow matching produces accurate results if complete monitoring information is provided. However, this is rarely the case in real traffic. When the monitoring provides information incomplete due to an overload, error, or encryption, the matching results are not satisfactory.

Table 5. The Jitter of Events From Flow Start and Flow End on the All-Information Dataset

Percentile	1	5	25	50	75	95	99
Event Earliness (ms)	0	0	1	2	3	5	8
Event Lateness (ms)	0	0	1	2	3	5	8

6.2 Time-Based Matching Method

Time-based event-flow matching is based on matching the *destination IP address*, *source IP address*, and *source port* similarly as the parameter-based matching method. It increases its matching accuracy by matching only those entities that occurred in a specific time-window. This method had been run on both the *all-information dataset* and the *reduced-information dataset*, where it produced the same results.

The first step in evaluating this matching method was the estimation of the size of the time-window. For all the related events and flows in the list of the confirmed event-flow pairs, we measured the jitter of the event time of occurrence from the flow start and from the flow end separately, to enumerate the *event lateness* and *event earliness*. As the DNS flow mostly consists of a single packet, the flow start and flow end times are equal; consequently, the *event lateness* and the *event earliness* are the same. The results may be observed in Table 5. Our results show that only 5 % of the related events happen at the same time as the DNS flow is captured. The possible reasons for this behavior are described in Section 3. On the other hand, most of the events are observed with at most 8 ms jitter from their related flows.

The 8 ms lower bound on the *event lateness* and the *event earliness* show the approximate size of the time-window to encompass most of the related events. However, it does not encompass all the events, and the larger the window is, the more probable it is that unrelated events and flows will be matched. The method had been run with different time-window settings on the *all-information dataset*, so the optimal size of the time-window could be found. The method's performance on these matching runs may be observed in Figures 2 and 3.

Figure 2 depicts the performance for small time-window sizes. The *accuracy* is close to one, as it is influenced by the vast amount of true negatives. *Precision* is precisely one, as there are no false positives for small time-windows. As the time-window expands, the number of false negatives decreases, and thus the *recall* and *F1-score* rise fast until they reach the 99 percentile (8 ms). The resolution of the graph then must be increased, as the changes for large time-windows are more subtle. Figure 3 shows that the first false positives appear at the 300 ms mark as the *precision* drops. The *F1-score* also drops, as the increase of false positives (*precision* falling) is not sufficiently compensated by the increase of true positives (*recall* rising). Consequently, for time-windows larger than 200 ms, the performance of time-based matching on this dataset falls steadily.

The time-based matching method with 200 ms time-window performed better on both datasets than the parameter-based (no qname) method. It missed a few related pairs, producing false negatives, but the *precision* reached one with no false positive pairs matched. Rated by the *F1-score*, this method came second-best on the *reduced-information dataset*, proving decent performance on incomplete monitoring data.

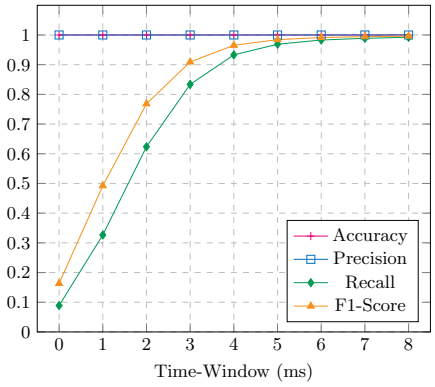

Fig. 2. Time-Based matching performance with small-sized time-window.

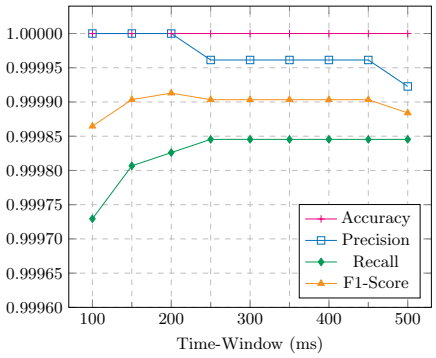

Fig. 3. Time-Based matching performance with large-sized time-window.

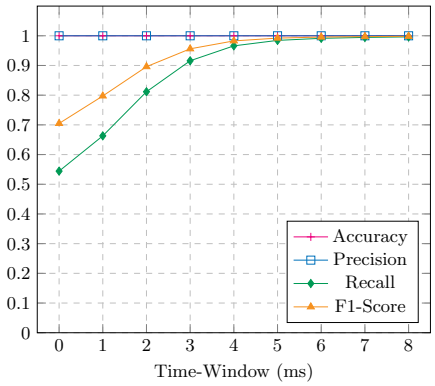

Fig. 4. Combined matching performance with small-sized time-window.

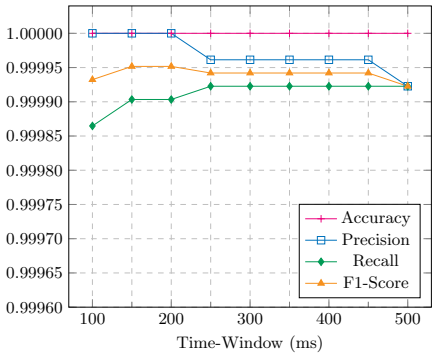

Fig. 5. Combined matching performance with large-sized time-window.

6.3 Combined Matching Method

The combined matching method uses the parameter-based matching where extended flow information is available and falls back to the time-based matching where it is not. The method had been run on both the *all-information dataset* and the *reduced-information dataset*. As the method used parameter-based matching on the the *all-information dataset*, it produced a list of event-flow pairs identical to the ground truth. The results on the *reduced-information dataset* are further explored.

Similarly to the time-based matching method, the combined method's performance with differently sized time-windows was examined on the small and large time-windows (Figures 4 and 5, respectively). The method behaves similarly as the time-based matching, but with a few advantages. The *recall* starts at a much higher value for the small windows, even with no toleration time-window (0 ms). This is caused by the number of flows the method had been able

to match based on their parameters only, not even considering the time-window. As a result, the *F1-score* rise is more gradual even for small time-windows. The *precision* and *F1-score* reached higher values for the large-sized windows than time-based matching, as it missed fewer related pairs. The measured *F1-score* implies that the lower and upper bound for the time-window are the same as for time-based matching, 8 ms, and 200 ms, respectively. Consequently, the optimal time-window remained the same at 200 ms.

The results of this method on the *all-information dataset* and the *reduced-information dataset* may be observed in Tables 3 and 4. The results on the *reduced-information dataset* make this method a promising candidate for matching events and flows in the real traffic, where monitoring errors and encrypted flows are common.

7 Lessons Learned

As shown by the event and flow capture model, the positioning of vantage points strongly influences the event-flow matching process. The flow monitoring probe must be positioned to capture all the flows that might invoke events on the event-monitored devices. Otherwise, there will be gaps in measured data and a high number of unmatchable events. The distance of the vantage points in the network and network features like latency, jitter, and congestion, also strongly influence the matching. Drop-outs and monitoring errors make the parameter-based event-flow matching perform poorly. On the other hand, distant vantage points and long latency negatively influence the time-based matching.

As our dataset shows, the observed network flows may be missing extended features. In these cases, the suitable approach is time-based matching. However, it requires the monitoring devices to be time-synchronized. In our dataset, all devices had been synchronized by the NTP with millisecond precision. This precision proved sufficient for a network with around 200 devices. With larger networks and higher levels of DNS activity, a microsecond precision might be needed.

8 Conclusion

According to the results of our experiment described in Section 6, the most suitable matching method for real network traffic is the method that combines parameter-based and time-based matching. It has the most accurate matching performance on the traffic with valid extended flow information. When supplied with data missing this parameter, e.g., the encrypted flows, it falls back to time-based matching. The time-based matching needs to be optimized to the network's specifics and does not perform as good as the parameter-based matching. However, the ability to match network flows without extended flow information makes it a suitable candidate for matching encrypted flows.

We were able to identify the queried domain name (*qname*) parameter in case of a successful match when the event-flow matching methods were run on the

encrypted DNS flows. This parameter is otherwise encrypted and unavailable to flow monitoring, so the event-flow matching may enrich flow monitoring with new data. With the rising amount of encrypted network traffic, event-flow matching might be the method to join the encrypted network flows with unencrypted information stored in event logs.

Acknowledgment

This research was supported by the CONCORDIA project that has received funding from the European Union's Horizon 2020 research and innovation programme under the grant agreement No 830927.

References

1. Brilingaitė, A., Bukauskas, L., Kutka, E.: Time-line alignment of cyber incidents in heterogeneous environments. In: ECCWS 2018 17th European Conference on Cyber Warfare and Security. p. 57. Academic Conferences and publishing ltd. (2018)
2. Bushart, J., Rossow, C.: Padding ain't enough: Assessing the privacy guarantees of encrypted DNS. In: 10th USENIX Workshop on Free and Open Communications on the Internet ({FOCI} 20) (2020)
3. Collins, M., Collins, M.S.: Network security through data analysis: building situational awareness. "O'Reilly Media, Inc." (2014)
4. Dreger, H., Kreibich, C., Paxson, V., Sommer, R.: Enhancing the accuracy of network-based intrusion detection with host-based context. In: International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. pp. 206–221. Springer (2005)
5. Haas, S., Sommer, R., Fischer, M.: Zeek-osquery: Host-network correlation for advanced monitoring and intrusion detection. arXiv preprint arXiv:2002.04547 (2020)
6. Henderson, S., Nicholls, B., Ehmann, B.: Time-based correlation of malicious events and their connections. `https://resources.sei.cmu.edu/asset_files/Presentation/2019_017_001_539987.pdf`, accessed: 2020-09-15
7. Houser, R., Li, Z., Cotton, C., Wang, H.: An investigation on information leakage of DNS over TLS. In: Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies. pp. 123–137 (2019)
8. Luque, A., Carrasco, A., Martín, A., de las Heras, A.: The impact of class imbalance in classification performance metrics based on the binary confusion matrix. Pattern Recognition **91**, 216–231 (2019)
9. Siby, S., Juarez, M., Diaz, C., Vallina-Rodriguez, N., Troncoso, C.: Encrypted DNS – privacy? A traffic analysis perspective. arXiv preprint arXiv:1906.09682 (2019)
10. Teng, S., Wu, N., Zhu, H., Teng, L., Zhang, W.: SVM-DT-based adaptive and collaborative intrusion detection. IEEE/CAA Journal of Automatica Sinica **5**(1) (2017)
11. Tovarňák, D., Špaček, S., Vykopal, J.: Traffic and log data captured during a cyber defense exercise. Data in Brief p. 105784 (2020)
12. Vasilomanolakis, E., Karuppayah, S., Mühlhäuser, M., Fischer, M.: Taxonomy and survey of collaborative intrusion detection. ACM Computing Surveys (CSUR) **47**(4), 1–33 (2015)

13. Velan, P., Čermák, M., Čeleda, P., Drašar, M.: A survey of methods for encrypted traffic classification and analysis. International Journal of Network Management **25**(5), 355–374 (2015)
14. Zhang, T., Zhu, Q.: Distributed privacy-preserving collaborative intrusion detection systems for VANETs. IEEE Transactions on Signal and Information Processing over Networks **4**(1), 148–161 (2018)
15. Špaček, S.: Enriching DNS flows with host-based events to bypass future protocol encryption - scripts for data processing. Zenodo (Oct 2020). https://doi.org/10.5281/zenodo.4064934