POSTER: A Semantic-aware Approach to Reasoning about Network Traffic Relations*

Hao Zhang Danfeng (Daphne) Yao Naren Ramakrishnan
Department of Computer Science, Virginia Tech
Blacksburg, VA, USA
{haozhang, danfeng, naren}@cs.vt.edu

ABSTRACT

This paper addresses the problem of reasoning about relations between network packets on a host or in a network. Our analysis approach is to discover the causal relations among network packets, and use the relational structure of network events to identify anomalous activities that cannot be attributed to a legitimate cause. The key insight that motivates our traffic-analysis approach is that higher-order information such as the underlying relations of events is useful for human experts' cognition and decision making. We design a new *pairing* method that produces special pairwise features, so that the discovery problem can be efficiently solved with existing binary classification methods. Preliminary experiments involving real world HTTP and DNS traffic show promising evidence of the accuracy of inferring the network traffic relations using our semantic-aware approach.

Categories and Subject Descriptors

C.2.0 [**General**]: Security and protection

Keywords

Network Security, Anomaly Detection, Classification

1. INTRODUCTION

This paper addresses the issue of reasoning about network traffic relations. We aim to develop a traffic monitoring tool that helps identify anomalous network traffic events, which may be caused by misconfigured hosts, infected hosts, or external attackers. The key insight that motivates our approach is that higher-level information such as the underlying relations or semantics of events is useful for human experts' cognition, reasoning, and decision making in cyber security [3]. Thus, analyzing relations among network events may provide important insights for identifying network anomalies. We focus on discovering the causal relations of network packets in this work.

*This work has been supported in part by NSF grants CAREER CNS-0953638 and CCF-0937133.

CCS'13, November 4–8, 2013, Berlin, Germany.

Most network analysis methods for filtering individual packets or aggregation statistics fail to fulfill this goal of event-relation discovery, with some exceptions. The most advanced solutions today aim to identify correlating traffic events or attributes (e.g., IP addresses). Gu *et al.* demonstrated the effectiveness of correlation analysis across multiple hosts of a network in detecting similarly infected bots [4]. King *et al.* constructed directed graphs from logs to show network connections for dissecting attack sequences [6]. These solutions mainly leverage domain knowledge of networked systems (e.g., protocol or system specifications) or attack behaviors (e.g., botnet command and control).

We present a new learning-based method and demonstrate with experiments that discovering fine-grained causality in network traffic is feasible. The causality of network packets provide contextual interpretations to the behaviors of systems and networks, illustrating why sequences of events occur and how they relate to each other. Because of the transitivity, the problem of discovering the packet dependencies among a set of events may be transformed into discovering the dependency of pairs of events, which we defined as *pairwise* relation.

General techniques for learning and recognizing directional causal relations of network traffic events do not exist. Most of the existing learning-based security studies are on binary classification problems, where an unknown instance (e.g., email, code, or network packet) needs to be classified into two classes – legitimate or suspicious. Our relation discovery work also differs from existing service dependency analysis (e.g., [1, 7]), because of *i)* our finer granularity (request vs. flow) and *ii)* different relation semantics.

2. THE PROPOSED APPROACH

The goal of our analysis is to infer the causality of the packets and infer the anomalies by enforcing specific security policy. The main operations in our analysis are Data Collection, Pairing, Data Labeling, Training, Classification, and Security Policy Enforcement. The Data Labeling, Training, and Classification operations are standard for machine learning based methods.

We introduce a new feature extraction method referred to as Pairing. This operation converts individual network events into event pairs with comparable pairwise attributes, so that conventional binary classification techniques can then be applied to discover the relations between packets. We also present the use of root-trigger policy in security policy enforcement to report anomalies.

A overview of our approach follows.

- DATA COLLECTION is intended to record and store the events to be analyzed. Each event e has one or more attributes $(A_1, \ldots, A_m)$ describing its properties.
- PAIRING is used to extract pairwise comparison results of events' attributes. Its inputs are two events $e = (A_1, \ldots, A_m)$ and $e' = (A'_1, \ldots, A'_m)$. The output is the event pair (e, e') with m pairwise attribute values $(B_1, \ldots, B_m)$, where a pairwise attribute $B_i (i \in [1, m])$ represents the comparison result of attributes A_i and A'_i. That is, $B_i = f_i(A_i, A'_i)$, where $f_i()$ is a comparison function (e.g., `isEqual`, `isGreaterThan`, `isWithinThreshold`, `isSubstring`, etc) for the type of the i-th attribute in the events.
- DATA LABELING is the operation that produces the correct causal relations for the event pairs in a small training dataset. We choose a binary label (1 or 0) to indicate the existence or non-existence of any causal relation in an event pair, e.g., $< (e, e'), 1 >$ represents that event e triggers e'.
- TRAINING is the operation that produces a machine learning model with labelled training data. We adopt two feature selection methods (*Information Gain* and *Gain Ratio*) to find an optimal set that could improve the effectiveness of machine learning classifiers.
- CLASSIFICATION is the operation to use the trained machine learning model to predict causal relations on new event pairs $\{P_{ij} = (e_i, e_j)\}$. We build and compare three common supervised machine learning classifiers – Naive Bayes, a Bayesian network, and a support vector machine (SVM) [2].
- SECURITY POLICY ENFORCEMENT is the operation intended to apply security policies to the classification results and report anomalous events.

2.1 Pairing Operation

The PAIRING operation extracts features of event pairs. An event attribute may be of the numeric, nominal, string, or composite type. Various comparison functions are chosen based on the attribute types.

- Numeric attributes can be compared by computing their difference, e.g., the interval `TimeDiff` between the timestamps of two network events, i.e. $B_i = A_i - A'_i$.
- A nominal attribute (e.g., file type, protocol type) categorizes the property of an event. Comparing nominal attributes usually involves string comparison, e.g., substring or equality tests.
- For the string type of attributes, we compute the similarity of the attribute values as the pairing attribute value. That is, $B_i = f_s(A_i, A'_i)$, where function f_s is a similarity measure, e.g., normalized edit distance. Taking HTTP packets as an example, we can compute a pairwise attribute `HostSim` by measuring the similarities between two host fields in the HTTP headers.
- A composite attribute contains multiple values, e.g., a destination address is composed of four octets for the IP address and an integer for the port. The comparison of two composite values is made by comparing the sub-attribute values separately. Therefore, we define a bitmap to store the comparison between the source and destination IPs. Each bit refers to the difference between one octet of two IPs or between two ports.

To reduce the complexity of the analysis, a heuristic is to pair up events whose timestamps are close. In our study, only the events whose timestamps differ by less than a certain threshold τ are compared. The purpose is to avoid making meaningless pairs for packets whose time difference is greater than τ.

2.2 Security Policy Enforcement

We use *root-trigger policy* as an example to explain the security policy enforcement. This policy is to determine whether the activity is legitimate or not based on the legitimacy of the initial cause of the event, i.e., the root trigger of the event. According to this definition, anomalous events are the events that do not have a valid root trigger. These events may be due to malware activities or host/server misconfiguration. A specific root-trigger security definition is based on user intention [8], where a valid root trigger should be related to a user activity (e.g., a function call to retrieve user inputs, mouse clicks, or keyboard inputs). In what follows, we refer to the events that do not have any valid root triggers as *vagabond* events.

Algorithm 1 Find-root Algorithm.

Input: $set = \{(e_i \rightarrow e_j)\}$ and an event e_k.
Output: a set $roots$, where each in $roots$ is a root of e_k.

```
1: define a set roots to store the results
2: define a queue Q and enqueue e_k onto Q
3: while Q ≠ ∅ do
4:    event n ← dequeue Q
5:    let a set ps ← all of n's parents
6:    for each event e ∈ ps do
7:       if e is of type root then
8:          roots = roots + {e}
9:       else if e ∉ Q then
10:         enqueue e onto Q
11:      end if
12:   end for
13: end while
14: return roots
```

Algorithm 1 finds all the roots of an event e_k, given all the pairwise causal relations. The input of the algorithm is a set of all the pairwise relations $\{(e_i \rightarrow e_j)\}$ and an event e_k. The output is a set containing all the roots of e_k. In order to compute the transitive reduction of a directed graph, we use a queue Q to perform breadth-first traversal.

The root-trigger security policy is suitable for identifying network activities that are not triggered by users, including but not limited to:

- Spyware exfiltrating sensitive information through outbound network traffic from the monitored host,
- Bots' command-and-control traffic, and attack activities (e.g., spam or DoS traffic) originated from the monitored host,
- Websites collecting and reporting user data.

One future direction is to systematically investigate the design and use of complex policies for network assurance.

3. PRELIMINARY EVALUATION

We use the Weka library [5] and implement the prototype in Java. The data we evaluate are summarized in Table 1 (τ is the threshold for the maximal time interval of a pair of events). Dataset I is composed of the user's events and outbound HTTP traffic that are sampled from a 20-participant user study. Each participant was asked to actively surf the web for 30 minutes on a laptop equipped with our data col-

lection program. We set the threshold as 30 seconds, as 97.2% of the HTTP requests fall within that interval range. Dataset II is obtained by using `tcpdump` to continuously collect outbound DNS queries and HTTP packets from a gradate student's workstation for 19 days. We collected types `A` and `AAAA` DNS queries and the packets containing `GET`, `HEAD`, or `POST` HTTP information. Due to the wide prevalence of DNS prefetching, we choose a relaxed threshold of 15 seconds, which covers 97.8% of the event pairs.

Data	Type	τ (s)	# of Pairs	Size (MB)
I	HTTP	30	572,725	38.61
II	DNS & HTTP	15	1,833,306	113.56

Table 1: An overview of datasets in the experiments.

Data labeling. Our rules for HTTP traffic are similar to what are used in [8]. We manually labeled 12% of HTTP requests in the training datasets, which also indicates the inadequacy of the existing rule-based approach. Rules for labeling mixed DNS and HTTP traffic involve analyzing the query of the DNS packet, type of DNS query(e.g., `A` or `AAAA`), host of the HTTP request, and the protocol version of destination IP address in HTTP header.

Classification. 10-fold cross-validation experiments demonstrate that the accuracy of both training sets is greater than 99%. Due to the sparsity of relations in network traffic, we define a cost matrix $C = \begin{bmatrix} 0,1 \\ 10,0 \end{bmatrix}$ that penalizes classifying false negative with 10, while penalizing classifying false positive only with 1. The binary classification accuracy for pairwise causal relations is consistently high for the Bayesian network and SVM methods (see Table 2). The naive Bayes classifier yields lower average accuracy, indicating that the conditional independence assumptions made by this classifier might be strong.

Data	# of pairs		**Naive Bayes**	**Bayesian Network**	**SVM**
	Training	Test			
I	309,921	262,804	99.77%	99.85%	99.92%
II	916,650	916,656	98.98%	100%	100%

Table 2: Pairwise classification accuracy results of train-n-test experiments. Sizes of training and test data are shown.

Security Policy Enforcement (Correctness of root triggers). Running the *find-root* procedure in Algorithm 1 on the pairwise classification results, we identify the root triggers of all events and compare them to the ground truth values.

By enforcing the root-trigger policy, the evaluation shows that for 99.0% of events, the roots are correct with respect to the ground truth. The results are the same for all three classifiers. Among the 99.0% events, we found 22 *vagabond* events, which belong to either malicious behavior (e.g., request to altfarm.mediaplex.com) or misconfiguration on the server (e.g., some requests to googleapi or twitter). There are 1.0% events whose root triggers are not correctly found. Manual investigation reveals that the wrong root triggers are all false positives, which are caused either by *i)* null or truncated attributes (e.g., referrer, hostname) due to the transmission issues, or *ii)* timestamps out of the specified threshold. The latter can be avoided by increasing the threshold in the pairing operation, which, however, may increase the computation overhead by generating unnecessary pairs.

Summary. The correctness of classifying the causal relation among network packets shows the feasibility and effectiveness of our analysis approach. The root-trigger security analysis allows us to identify network events linked to malicious hosts or due to misconfiguration of web servers. The classification performance is efficient in general and adequate for fast traffic analysis.

4. CONCLUSIONS AND FUTURE WORK

Our learning-based technique to discover causal relations shows promising application to analyzing host-based outbound HTTP and mixed HTTP and DNS traffic data. Our experiments identified several types of network anomalies caused by traffic to malicious servers or misconfigured servers.

Future work will proceed along three directions. First, we plan to explore the incorporation of more complex security policies, which could help detect more types of threats. Second, we plan to explore the inference of more complex boolean relationships across events [9] than simple pairwise relations (e.g., at least two of three precursors must be present for a given event). Finally, we intend to encapsulate the entire framework, from pairwise relation construction, to classification, in a machine learning framework so that all necessary parameters can be jointly optimized. Such an approach can also help avoid making arbitrary thresholding decisions and better explore the joint interplay between design decisions.

5. REFERENCES

[1] X. Chen, M. Zhang, Z. M. Mao, and P. Bahl. Automating network application dependency discovery: Experiences, limitations, and new solutions. In *Proceedings of OSDI*, pages 117–130, 2008.

[2] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[3] T. Green, W. Ribarsky, and B. Fisher. Visual analytics for complex concepts using a human cognition model. In *Proc. IEEE VAST*, pages 91 – 98, October 2008.

[4] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium*, 2008.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[6] S. T. King, Z. M. Mao, D. G. Lucchetti, and P. M. Chen. Enriching intrusion alerts through multi-host causality. In *Proceedings of Network and Distributed System Security (NDSS)*, 2005.

[7] A. Natarajan, P. Ning, Y. Liu, S. Jajodia, and S. E. Hutchinson. NSDMiner: Automated discovery of network service dependencies. In *INFOCOM*, pages 2507–2515, 2012.

[8] H. Zhang, W. Banick, D. Yao, and N. Ramakrishnan. User intention-based traffic dependence analysis for anomaly detection. In *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*, pages 104–112. IEEE, 2012.

[9] L. Zhao, M. J. Zaki, and N. Ramakrishnan. Blosom: a framework for mining arbitrary boolean expressions. In *Proc. KDD'06*. ACM, 2006.