Full-Stack Information Extraction System for Cybersecurity Intelligence

Youngja Park and Taesung Lee
IBM T. J. Watson Research
Yorktown Heights, NY 10598, USA
young_park@us.ibm.com, taesung.lee@ibm.com

Abstract

Due to rapidly growing cyber-attacks and security vulnerabilities, many reports on cyber-threat intelligence (CTI) are being published daily. While these reports can help security analysts to understand on-going cyber threats, the overwhelming amount of information makes it difficult to digest the information in a timely manner. This paper presents, SecIE, an industrial-strength full-stack information extraction (IE) system for the security domain. SecIE can extract a large number of security entities, relations and the temporal information of the relations, which is critical for cyberthreat investigations. Our evaluation with 133 labeled threat reports containing 108,021 tokens shows that SecIE achieves over 92% F1-score for entity extraction and about 70% F1-score for relation extraction. We also showcase how SecIE can be used for downstream security applications.

1 Introduction

A rapid increase in cyberattacks, both in number and attack techniques, poses enormous challenges to security analysts. Much of the information on new threats often appear first in unstructured reports such as blogs and news articles. To quickly respond to the on-going attacks, it is critical to digest the information about new threats in a short period of time. However, it is very difficult to find relevant information from CTI reports, particularly because cyber-attacks involve many different entities, including the attacker, victim (e.g., companies/industries), tools (e.g., malware) indicators of compromise (IOCs, e.g., file names and IP addresses), and various relations, some of which may be unknown to the security experts.

We present a large-scale full-stack IE system designed for the cybersecurity domain. SecIE can extract 26 entity types, 20 fixed relation types and various Open IE relations, and the time information of the relations, which is very critical in cybersecurity. Figure 1 shows a snippet of a CTI report[1] and the IE results from SecIE. The entity extraction model detects mentions of *Malware* and *ThreatActor* from the text. The coreference resolution model recognizes that 'the new ransomware' refers to Trojan-Ransom.Win32.Xpan and 'a gang' refers to TeamXRat. Finally, the relation extraction module produces a relation tuple, *<Trojan-Ransom.Win32.Xpan, developed_by, TeamXRat>*, from "the new ransomware was developed by a gang".

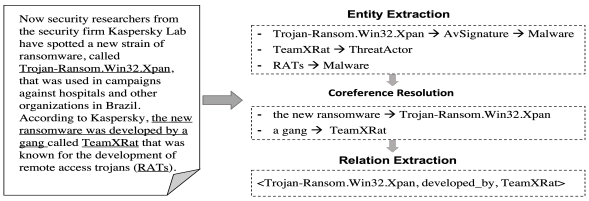


Figure 1: A CTI report and the security entities and relation extracted by SecIE

While there have been efforts to apply NLP and IE to the cybersecurity domain (Joshi et al., 2013; Lal, 2013; Jones et al., 2015; Bridges et al., 2017; Liao et al., 2016; Husari et al., 2017; Pingle et al., 2019; Yi et al., 2020), they target on a specific sub-area of cybersecurity, mostly on extracting IOCs or vulnerabilities, or a component (either entity extraction or relation classification) in the IE process. To our knowledge, our system is the largest end-to-end IE system for the cybersecurity domain supporting a large number of security entity and relation types.

Most existing IE systems apply supervised (deep) learning methods relying on a large

[1] https://www.cyberdefensemagazine.com/teamxrat-spreads-ransomware-via-rdp-brute-force-attacks/

Proceedings of EMNLP 2022 Industry Track, pages 541–549
December 9–11, 2020.

amount of high-quality labeled data. Unlike the general domain types, labeling fine-grained security entities and relations requires deep domain knowledge, and, thus it is much more difficult to produce a high-quality training data for the security domain. As an anecdote, 3 annotators (1 security expert and 2 professional annotators with many years' experience) working full-time for 5 months could produce only 133 annotated documents, which are far from enough to train supervised models for our need. Thus, SecIE applies unsupervised NLP technologies. We develop tehcniques to handle idiosyncrasies in security terms and and take into account the structural characteristics found in many CTI reports. This domain customization allows SecIE highly accurate, achieving over 92% F1 for entity extraction and 70% F1 for relation extraction.

2 Methodology

We employ a pipeline architecture as shown in Figure 2, consisting of document parsing; linguistic analysis; entity extraction; coreference resolution; topic entity detection; relation extraction; and relation time assignment. Input

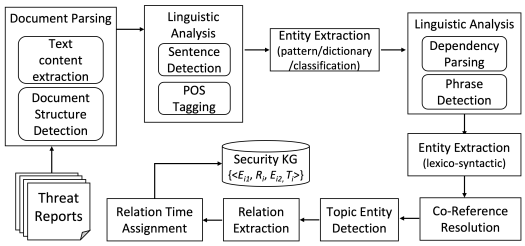


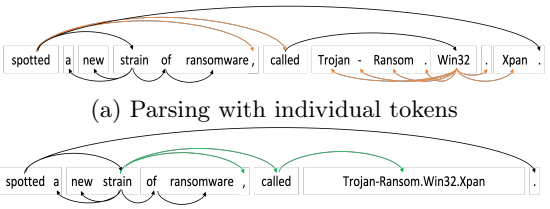
Figure 2: High-level System Architecture

documents are processed sequentially, where the document content and all the results from the previous components are passed as input to the next component. However, the system can process multiple documents in parallel yielding a high throughput.

2.1 Document Pre-processing

Document Parsing: This component performs text content extraction and document structure detection. We use Apache Tika[2] to extract the file content and structural information such as titles, hyperlinks, tables, and list structures from the input files. The extracted structures are stored as annotations over the document content and passed to the subsequent components along with the content.

Linguistic Analysis: This component performs sentence boundary detection, part-of-speech (POS) tagging and dependency parsing. We use SyntaxNet (Andor et al., 2016) for POS tagging and dependency parsing. It was trained with general domain documents and often fails to parse security sentences correctly, because some security entities include many tokens and punctuation marks internally (e.g., some URLs have over 100 tokens). To improve the parsing accuracy, we first detect entity mentions and pass the entire entity mention as a noun token to the parser. Figure 3 shows a sample sentence and the parsing results when all tokens are passed to the parser individually and when entity mentions are passed as a token.



(a) Parsing with individual tokens

(b) Parsing with entities as tokens

Figure 3: Improved sentence parsing through domain customization

2.2 Entity Extraction

We identified the 26 fine-grained entity types related to malware, IoCs, and security vulnerability. The types are determined based on the STIX standard[3] which defines 9 key security concepts and their relationships. The full list of our entity types are shown in Figure 6 in Appendix. We provide a type inheritance as shown in Figure 6, allowing applications to consume the entity types at different levels.

Pattern-based Method is used for entity types with well-defined patterns such as *IPAddress* and *EmailAddress*. We note that many CTI reports, especially those published online, often use obfuscated forms for malware IOCs, such as 'X.Y.177.245','82(dot)103(dot)137(dot)14','BLOCKED.BLOCKED.172.196',and'x0x0.[REMOVED].com.br'. Thus, SecIE supports many obfuscated IOC patterns, unlike other existing tools.

[2]https://tika.apache.org

[3]https://oasis-open.github.io/cti-documentation/

Dictionary-based Method is used when a reputable list of terms belonging to a certain entity type exists. In the cybersecurity domain, previously known *Campaign*, *Malware* and *ThreatActor* cases are well documented. In these cases, we match the dictionary terms with the noun phrases. However, this dictionary matching method can extract only previously known samples. We address this problem using the lexico-syntactic pattern matching method to extract new mentions.

Lexico-Syntactic Pattern-based Method Inspired by the findings in (Hearst, 1992), we apply the following syntactic patterns to extract security entities: (1) NP (, NP)* BE NP; (2) NP, CALLED NP; (3) NP such as NP (, NP)* (4) NP including NP (, NP)*; (5) NP a.k.a | ((which|that)? (BE)? (also)? CALLED as) NP. Here, NP stands for a noun phrase, BE represents the be-verbs (e.g., 'is'), and CALLED includes 'dubbed', 'called', 'named', 'known', 'referred', and 'termed'.

To discover new mentions, we first check if the entity type of an NP in these syntactic patterns is determined. Then, we label the remaining NPs to the same entity type. If the types of multiple NPs are determined, they should be the same type or have a super-subtype relation. We also use a predefined set of cue words to detect new mentions for *Campaign* and *Malware*, and *ThreatActor* (Table 6 in Appendix). If a cue word matches with NP or NP's headword, we classify the other NPs to the same entity type as the cue word. Table 1 shows sample sentences where 'WannaCry', 'Wcrypt', 'WCRY', 'WannaCrypt' are extracted as *Malware* even though the mentions were unknown.

1) WannaCry is a ransomware worm that spread rapidly ...
2) A new ransomware dubbed "WannaCry" is ...
3) The WannaCry ransomware has been very active since ...
5) WannaCry, also known as Wcrypt, WCRY, WannaCrypt

Table 1: Examples of new mention extraction. The numbers indicate the rule used to determine the entity type.

Classification-based Extraction *AvSignature* mentions do not conform to particular patterns making regex-based method ineffective (e.g., 'ADWARE/Agent.imv', 'Trojan-Ransom.Win32.Xpan'). Further, the number of *AvSignature* instances is very large (millions), making the dictionary method inefficient. However, they have distinct word shapes which are very different from regular words, and it is easy to collect many examples from public sources. We collected 660,000 *AvSignature* names from *VirusTotal* as the positive sample and added 470,000 words randomly chosen from CTI reports as the negative sample. We then trained a Logistic Regression model using character n-gram and word shape features (e.g., uppercase/lowercase letters, digits and symbols).

2.3 Coreference Resolution

We categorize coreferences into two types based on the search range for the referent.

Within-sentence Coreference appears in certain syntactic structures that connect two noun phrases, such as appositives, relative pronouns (*e.g.*, 'which'), or certain phrases such as "<*nominative noun*> [,] CALLED [as] <*proper noun*>". When the proper noun belongs to a security entity, we resolve the nominative noun or pronoun to the proper noun. Figure 1 shows two examples of witin-sentence coreferences: "a new strain of ransomware, called Trojan-Ransom.Win32.Xpan" and "a gang called TeamXRat". We resolve "a new strain of ransomware" to "Trojan-Ransom.Win32.Xpan" and "a gang" to "TeamXRat".

Cross-sentence Coreference Syntactic analysis alone cannot connect two mentions together when they appear in different sentences. We use a document structure-based sentence embedding model proposed in (Lee and Park, 2019), which generates semantic representations for sentences using BERT (Devlin et al., 2019; Joshi et al., 2019). If a sentence contains a nominative or pronoun mention (e.g., 'the malware'), we identify semantically related sentences for the sentence based on the sentence embeddings and find its referent from the proper nouns in the related sentences. We replace the nominative or pronoun mention with each of the candidates, calculate the likelihood of the candidate in the sentence, and choose the candidate with the highest likelihood as the referent.

2.4 Topic Entity Detection

Most CTI reports provide a deep analysis on a particular malware or campaign. We call the focus of a CTI report the topic entity. Many

CTI reports are very succinct, often simply providing the list of related entities, such as IOCs, without contextual connection to the topic entity. These related entities provide critical intelligence about the topic entity, and connecting them with the corresponding malware or campaign is critical. We identify the topic entity of CTI reports as follows. We first look for mentions of *Malware*, *Campaign*, and *ThreatActor* in the first 15 sentences. When there are multiple mentions of these types, we choose the topic entity based on the following factors: (1) the position of the sentence (likely to appear early in the article); (2) if the mention is a singular or plural (tend to be singular); (3) the syntactic role of the mention in the sentence (likely the subject or the object); (4) the occurrence count of the mention in the article (likely to appear many times).

2.5 Relation Extraction

Similarly to entity extraction, we apply several different techniques for relation extraction.

OpenIE Relation Extraction discovers relations from certain syntactic structures (Angeli et al., 2015; Banko et al., 2007; Soderland et al., 2010; Fader et al., 2011; Mausam et al., 2012; Roy et al., 2019). Many security relations involve actions (e.g., download, connect, etc.). Thus, we focus on the three syntactic structures containing a verb phrase and two noun phrases: <NP(subj)-VP-NP(obj)>, <NP-VP-pp-NP>, and <VP-NP-pp-NP>, where pp is a preposition. We find these syntactic structures in sentences, and, if both NP arguments are security entity mentions, we extract a relation by associating the NPs with the VP as the relation type. Table 2 shows examples of semantic relations extracted using this method.

Cooccurrence-based Relation Extraction While the OpenIE relations provide useful semantic relations, extracting relations only from the three structures can miss other relevant relations. We generate relations if two security entities co-occur in a sentence but are not connected by an OpenIE relation. The assumption is that if the two entities frequently appear together in the same sentence, they should be of interest to security analysts. We produce cooccurrence-based relations between the five main security entities: *Campaign*, *Indicator*, *Malware*, *ThreatActor* and *Vulnerability* and assign a generic relation type ('related'). Table 3 shows sample co-occurrence-based relations.

Relations with Topic Entity As discussed above, many threat reports describe information about a particular security event or entity, and other entities in the document provide insights on the topic entity. In this work, if the entities in a list are not included in any other relations, we connect them to the topic entity via a relation type denoted as *related+EntityType* (*e.g.*, relatedHash).

2.6 Temporal Information Extraction

Threat intelligence is time sensitive, and knowing when a security event has occurred is critical. Time information can be expressed in multiple ways, including point-in-time (e.g., "2016-05-25"), relative time (e.g., "last year"), time range (e.g., "2016–2017"), and embedded time (e.g., "CVE-2017-3018"). `SecIE` extracts these time expressions and normalize them to the timestamp. For relative time expressions, we infer their point-in-time based on an anchor time, which can be an absolute time expression in nearby sentences. If there are no point-in-time expressions in the document, we use the file's last modified time or the publication date as the anchor time. Then, we use the following priority orders to determine which temporal information gets assigned to a relation: (1) time in the same dependency construct; (2) time in the same sentence (3) time in the previous sentences; (4) the document's last modified time; and (5) The document's published time Figure 7 in Appendix shows a sample threat report and the output of `SecIE` including the entity, relation, and time information.

3 Performance Evaluation

To evaluate our system, we manually labeled 133 CTI reports, which contain 6,438 sentences and 108,021 tokens. The documents were labeled by 3 full time annotators over 5 months. To ensure the quality of the labeled data, we kept only the labels agreed by all 3 annotators, resulting in 3,295 entity and 1,216 relation mentions. More detailed statistics of the annotations are shown in Table 7 and Table 8 in Appendix.

Extracted Relation	Input Sentence
<Locky, spread_through, Necurs botnet>	Locky ransomware is again being spread through the Necurs botnet.
<zhCat, listen_on, port 1000>	If the attackers set up a zhCat instance listening on port 1000 on 192.168.116.128 ...
<zhCat, listen_on, 192.168.116.128>	
<Shamoon, deloyed_on, Saudi Aramco>	the Iranians deployed the Shamoon malware on Saudi Aramco, ...

Table 2: Examples of OpenIE relation extraction

Extracted Relation	Input Sentence
<Dyre variants, related, win32k.sys>	New Dyre variants exploiting CVE-2015-0057, a use-after-free vulnerability in the win32k.sys component
<KaiXin EK, related, 125.77.31.181>	125.77.31.181 port 12113 – otc.szmshc.com:12113 – KaiXin EK
<KaiXin EK, related, otc.szmshc.com:12113>	

Table 3: Examples of occurrence-based relation extraction

3.1 Entity Extraction Results

Table 4 shows the performance of our entity extraction (see Table 9 and Table 10 in Appendix for the performance for all entity types). The evaluation is performed by measuring the mention-level precision (P), recall (R) and F1 scores over all entity types. SecIE$_{all}$ reports the performance for all 133 reports, showing that SecIE achieves a very high F1 score with a good balance between precision and recall.

Further, we compare SecIE with a deep learning model to illustrate the challenges for applying supervsied learning methods for cybersecurity data. We split the 133 labeled documents into train (80%), validation (10%) and test (10%) datsets, consisting of 106, 14 and 13 documents respecitively, and trained a BERT model as described in (Devlin et al., 2019). The results ($_{small}$) validate that SecIE significantly outperforms the BERT model.

Model	Precision	Recall	F1
SecIE$_{all}$	95.1	89.4	92.2
SecIE$_{small}$	89.8	84.7	87.2
$BERT_{small}$	83.3	70.3	76.2

Table 4: Performance of entity extraction models. *all* denotes the 133 labeled documents, and *small* denotes the 13 test dataset.

3.2 Relation Extraction Results

We measure the performance of relation extraction using four different settings.

- *ExactMatch*: An extracted relation and the ground truth must have the same entity spans, entity types and the relation type.
- *-eType*: The condition for the entity type match is removed from *ExactMatch*. This is mainly because *Malware* and *Campaigin* are often interchangeably used.
- *-rType*: The condition for the relation type match is removed from *ExactMatch*.
- *-eType-rType*: Both the entity type and the relation type can be different.

Further, we evaluate the performance of relation extraction with and without co-reference resolution to show the effectiveness of the co-reference resolution step. Table 5 shows the evaluation results demonstrating SecIE's effectiveness. It produces over 70% precision across all settings, and co-reference resolution improves the performance, especially the recall.

	Without Coref.			With Coref.		
	P	R	F1	P	R	F1
ExactMatch	70.5	65.0	67.6	70.1	65.5	67.7
-eType	72.7	66.9	69.7	72.6	67.8	70.2
-rType	72.9	65.6	69.1	72.3	66.0	69.0
-eType-rType	75.9	67.8	71.6	75.7	68.8	72.1

Table 5: Relation extraction performance using different matching strategies and coreference settings.

4 Security Applications

We demonstrate how SecIE can provide additional insights on security incidents.

4.1 Malware Analysis

SecIE can be used to build a knowledge graph (KG) on malware from text. Figure 4 shows an input document about WannaCry[4] and the output KG. As we can see, SecIE extracted all of the security entities and connected them to the topic entity (*new variant of WannaCry*).

4.2 Inconsistency in CVEs

The NVD (national vulnerability database) provides information about known security vulnerabilities including the descriptions and asso-

[4]https://www.cybereason.com/cybereason-reveals-a-new-variant-of-wannacry-ransomware/

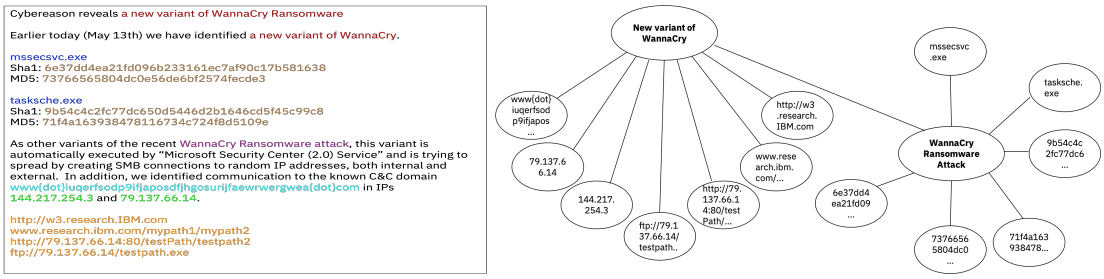


Figure 4: A KG built by SecIE from a report about the *WannaCry* ransomware

ciated metadata generated by domain experts. Even though the metadata was carefully curated by human, it can still contain errors. In particular, the affected software and the versions mentioned in the textual description and metadata can be different as shown in Figure 5. These inconsistencies can cause a harm, as many security applications rely on the metadata to identify vulnerable products in their environemnt.

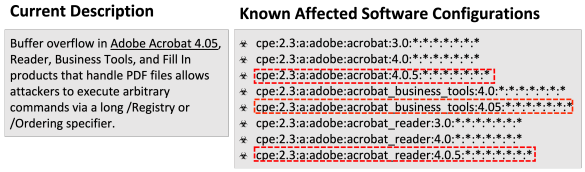


Figure 5: Example of inconsistent CVE

We match the mentions of *Application* extracted from the description and the CPE entries in the metadata using simple matching rules. Since an application can be referred by several synonyms (*e.g.*, Microsoft Office vs. Office), we apply a loose matching for application names. The versions can be represented as an exact version (*e.g.*, 4.05), a range (*e.g.*, 'before 10.3'), or wildcard symbols (*e.g.*, 4.x or 4.*), so we match the versions accordongly.

We randomly selected 168 CVE records and manually checked the inconsistency check results. This technique detected 26 potential inconsistencies, and 6 of them were confirmed to be inconsistent. This demonstrates that our tool can be used to find potentially errorneous CVE records and help to improve the quality of the CVE database.

5 Related Work

There have been a few efforts to apply IE to the cybersecurity domain. Most existing works focus on entity extraction for a small number of security entities (mainly, IOCs and Vulnerabilities) from certain security text (mainly, CVEs and Tweets). Joshi et al. (Joshi et al., 2013) present a system that produces linked data from CVE records. This system can extract 6 entity types commonly found in CVEs and link the extracted instances to DBPedia entries. (Sabottke et al., 2015) proposes a Twitter-based exploit detector, which collects tweets mentioning vulnerabilities. This tool uses a simple keyword matching and monitors occurrences of the "CVE" keywords and IDs in tweets. Liao *et al.* (Liao et al., 2016) presents a system (iACE) for fully automated IOC extraction. iACE detects file name, IP address, and URL using regular expressions. TTPDrill (Husari et al., 2017) extracts threat actions (*i.e.*, TTP) from security reports and map them to a threat action ontology from ATT&CK and CAPEC. This tool detects threat actions from the SVO dependency structure, where the subject is a malware instance. (Yi et al., 2020) presents an NER tool for the cybersecurity domain, which is similar to our entity extraction component. They apply regular expressions, dictionary matching and a CRF classifier for about 20 different entity types and achieves about 82% F1 score.

6 Conclusion

We presented a large-scale full-stack IE system developed for the cybersecurity domain. Through careful design choices to handle the idiosyncrasies in the cybersecurity data, our system achieves high F1 scores for both entity extraction and relation extraction. We also demonstrated how our system can be used for downstream applications. Our system can help security analysts by transforming the unstructured threat reports into structured formats which can be easily consumable by subsequent security applications.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *ACL 2016*.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *ACL 2015*.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI*.

Robert A. Bridges, Kelly M.T. Huffer, Corinne L. Jones, Michael D. Iannacone, and John R. Goodall. 2017. Cybersecurity automated information extraction techniques: Drawbacks of current methods, and enhanced extractors. In *The 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992*.

Ghaith Husari, Ehab Al-Shaer, Mohiuddin Ahmed, Bill Chu, and Xi Niu. 2017. Ttpdrill: Automatic and accurate extraction of threat actions from unstructured text of CTI sources. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, pages 103–115. ACM.

Corinne L Jones, Robert A Bridges, Kelly MT Huffer, and John R Goodall. 2015. Towards a relation extraction framework for cyber-security concepts. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference*.

Arnav Joshi, Ravendar Lal, Tim Finin, and Anupam Joshi. 2013. Extracting cybersecurity related linked data from text. In *2013 IEEE Seventh International Conference on Semantic Computing*, pages 252–259. IEEE Computer Society.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *EMNLP 2019*.

Ravendar Lal. 2013. Information Extraction of Security related entities and concepts from unstructured text. Master's thesis, May.

Taesung Lee and Youngja Park. 2019. Unsupervised sentence embedding using document structure-based context. In *ECML-PKDD 2019*, pages 633–647.

Xiaojing Liao, Kan Yuan, XiaoFeng Wang, Zhou Li, Luyi Xing, and Raheem A. Beyah. 2016. 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*, pages 755–766.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 523–534.

Aditya Pingle, Aritran Piplai, Sudip Mittal, Anupam Joshi, James Holt, and Richard Zak. 2019. Relext: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*.

Arpita Roy, Youngja Park, Taesung Lee, and Shimei Pan. 2019. Supervising unsupervised open information extraction models. In *EMNLP-IJCNLP 2019*.

Carl Sabottke, Octavian Suciu, and Tudor Dumitras. 2015. Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 1041–1056.

Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Mausam, and Oren Etzioni. 2010. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102.

Feng Yi, Bo Jiang, Lu Wang, and Jianjun Wu. 2020. Cybersecurity named entity recognition using multi-modal ensemble learning. *IEEE Access*, 8:63214–63224.

A Appendix

A.1 Target Entity Types

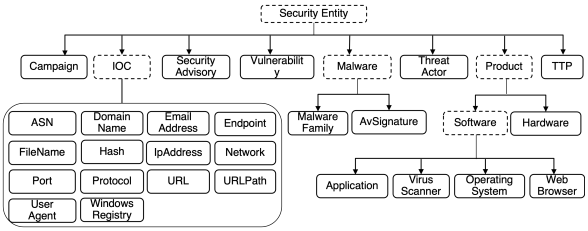


Figure 6: Target entity types and the type hierarchy in SecIE

A.2 Entity Cue Words

Entity Type	Cue Words
Campaign	breach, campaign, cyber attack, espionage, hack, scam
Malware	botnet, adware, crimeware, malware, ransomware, payload, RAT, spyware, trojan, virus, worm
ThreatActor	APT group, attacker, cyber criminal, cybercrime team, hacker, hacking group, malicious group, threat actor

Table 6: Examples of cue words for *Campaign* and *Malware*, and *ThreatActor*

A.3 Relations with Temporal Information

Petya Ransomware Spreading Via EternalBlue Exploit

On June 27, 2017, multiple organizations – many in Europe – reported significant disruptions they are attributing to a variant of the Petya ransomware. Based on initial information, this variant of the Petya ransomware may be spreading via the EternalBlue exploit used in the WannaCry attack from last month.

.....

FireEye has confirmed the following two samples related to this attack:

- 71b6a493388e7d0b40c83ce903bc6b04
- e285b6ce047015943e685e6638bd837e

Arg_1	Relationship	Arg_2	Time
Petya	spread_via	EternalBlue	2017-06-27
Petya	related	WannaCry	2017-05-27
EternalBlue	used_in	WannaCry	2017-05-27
Petya	relatedHash	71b6a493388e...	2017-06-27
Petya	relatedHash	e285b6ce0470...	2017-06-27

Figure 7: Example of relation extraction with temporal information

A.4 Details of the Experimental Data

Entity Type	Count
Vulnerability	805
MalwareFamily	682
TTP	540
FileName	276
URL	239
DomainName	216
AvSignature	126
ThreatActor	105
Hash	101
SecurityAdvisory	94
Campaign	43
IpAddress	40
WindowsRegistry	10
EmailAddress	8
Endpoint	7
Network	3
Total	3,295

Table 7: Distribution of Entity Types

Relation Type	Count
Cooccurrence relation	588
OpenIE relations	306
relatedFileName	73
relatedMalware	65
relatedDomainName	45
related URL	35
relatedHash	26
relatedVulnerability	26
relatedThreatActor	11
relatedCampaign	10
relatedWindowRegistry	9
relatedCnC	6
relatedEndpoint	6
relatedIpAddress	6
relatedNetwork	2
relatedUserAgent	2
Total	1,216

Table 8: Distribution of Relation Types

A.5 Entity Extraction Performance

Entity Type	P	R	F1
AvSignature	91.9	90.5	91.2
Campaign	75.9	95.3	84.5
DomainName	91.7	91.7	91.7
EmailAddress	85.7	75.0	80.0
Endpoint	87.5	100.0	93.3
FileName	88.1	88.4	88.2
Hash	100	100	100
IpAddress	97.5	97.5	97.5
MalwareFamily	96.0	83.7	89.4
Network	100	100	100
SecurityAdvisory	96.2	54.3	69.4
ThreatActor	100	77.1	87.1
TTP	94.4	84.3	89.0
URL	97.5	98.3	97.9
Vulnerability	97.9	98.3	98.1
WindowsRegistry	100	100	100
Average	95.1	89.4	92.2

Table 9: Performance of the entity extraction models by entity types

Entity Type	BERT	SecIE
AvSignature	70.59	100.00
Campaign	66.67	100.00
DomainName	93.75	100.00
EmailAddress	0.00	66.67
Endpoint	-	-
FileName	90.20	88.00
Hash	100.00	100.00
IpAddress	100.00	100.00
MalwareFamily	62.92	91.11
Network	-	-
SecurityAdvisory	96.55	98.31
ThreatActor	22.2	93.33
TTP	67.20	65.31
URL	90.91	96.30
Vulnerability	73.42	91.82
WindowsRegistry	-	-

Table 10: Comparison of a BERT model and SecIE on 13 test documents