Contents lists available at ScienceDirect

Engineering

journal homepage: www.elsevier.com/locate/eng

Research
Cybersecurity—Article

A Practical Approach to Constructing a Knowledge Graph for Cybersecurity

Yan Jia, Yulu Qi, Huaijun Shang, Rong Jiang, Aiping Li *

School of Computer Science, National University of Defense Technology, Changsha 410073, China

ARTICLE INFO

Article history:
Received 10 December 2017
Revised 21 December 2017
Accepted 7 January 2018
Available online 9 February 2018

Keywords:
Cybersecurity
Knowledge graph
Knowledge deduction

ABSTRACT

Cyberattack forms are complex and varied, and the detection and prediction of dynamic types of attack are always challenging tasks. Research on knowledge graphs is becoming increasingly mature in many fields. At present, it is very significant that certain scholars have combined the concept of the knowledge graph with cybersecurity in order to construct a cybersecurity knowledge base. This paper presents a cybersecurity knowledge base and deduction rules based on a quintuple model. Using machine learning, we extract entities and build ontology to obtain a cybersecurity knowledge base. New rules are then deduced by calculating formulas and using the path-ranking algorithm. The Stanford named entity recognizer (NER) is also used to train an extractor to extract useful information. Experimental results show that the Stanford NER provides many features and the useGazettes parameter may be used to train a recognizer in the cybersecurity domain in preparation for future work.

1. Introduction

At present, some cybersecurity knowledge bases have already been established for certain aspects of the cybersecurity domain. For example, common vulnerability enumeration is a vulnerability database in which each vulnerability has been given a unified ID, as defined by the MITRE Corporation. Other information is also available, such as threat level, threat type, and so on. A process knowledge base provides basic information about common processes, and some well-known antivirus vendors have established enormous signature bases about viruses. In addition, the Internet has become a primary source of knowledge and information [1], and contains a great deal of cybersecurity-related content such as security blogs, hacker forums, and security bulletins. Making full use of cybersecurity-related information from a variety of knowledge bases and websites, and putting all this security-related knowledge together, will be helpful for intrusion detection and cybersecurity situational awareness.

The main work in this paper is divided into two parts. The first part discusses building a cybersecurity knowledge base following a three-step procedure, and proposes a framework to build the knowledge base: First, obtain information by collecting and analyzing structured data and unstructured data; second, construct the ontology according to the information that has been obtained; and third, generate the cybersecurity knowledge base. The second part discusses cybersecurity knowledge deduction. A quintuple model of a cybersecurity knowledge base is proposed in order to obtain new knowledge using the path-ranking algorithm.

The quintuple cybersecurity knowledge base model [2] proposed in this paper contains the following five elements: *concept*, *instance*, *relation*, *properties*, and *rule*. This model provides a foundation for ontology construction. Machine learning is used to extract cybersecurity-related entities because the conditional random field model is suitable for named entity recognition. This paper uses the Stanford named entity recognizer (NER) to extract cybersecurity-related entities, and uses the Stanford NER base implementation to train an extracting model in the cybersecurity domain. To verify the influence of useGazettes, we built three different models. The experimental results show that useGazettes is important as a means of training an NER in the cybersecurity domain. Knowledge deduction includes attribute deduction and relationship deduction. For attribute deduction, the new attribute can be obtained using the attribute value prediction formula. For relationship deduction, the new relationship between instances can be obtained using the relational reasoning predictive formula and the path-ranking algorithm.

* Corresponding author.
E-mail address: liaiping@nudt.edu.cn (A. Li).

https://doi.org/10.1016/j.eng.2018.01.004

The rest of this paper is organized as follows: Section 2 discusses related work, and Section 3 presents our framework in detail. Section 4 provides a knowledge deduction scheme, and Section 5 provides a conclusion and suggestions for future work.

2. Related works

2.1. Ontology construction

A significant work was completed by Undercoffer et al. [3] from the University of Maryland, who developed an ontology to model attacks and related entities. The proposed ontology is only aimed at attack. In order to represent concepts and entities that are relevant to the cybersecurity domain, Joshi et al. [4] proposed an ontology for cybersecurity derived from the ontology proposed by Undercoffer. They extended the ontology to provide model relations that capture the US National Vulnerability Database (NVD) schema structure and security exploit concepts. This ontology contains 11 entity types (e.g., vulnerability, product, means, consequence). More et al. [5] extended the ontology proposed by Undercoffer and added rules to the reasoning logic. Their ontology comprises three fundamental classes: means, consequences, and targets.

In addition, a corporation named MITRE has been investigating ways to develop an ontology for the cybersecurity domain [6,7]. MITRE has created several standards and datasets for specific topic areas within the cybersecurity domain. A rich database provides the greatest advantage. Based on the efforts of Undercoffer and MITRE, Iannacone et al. [8] proposed an ontology for a cybersecurity knowledge base. This ontology represents the result of an iterative design process aimed at creating a knowledge representation that can effectively combine data from as many sources as possible within the cybersecurity domain. The resulting ontology contains 15 entity types and 115 properties in total.

2.2. Information extraction

The technology of information extraction has drawn increasing attention. At present, there are two main methods of knowledge extraction.

The first main method is based on knowledge engineering; it relies heavily on extraction rules, but can allow the system to handle information-extraction problems in specific domains. Most early information-extraction systems are based on extraction rules. The downside is that both domain-related professionals and linguists are required to participate in the development of the system. Because of the high extraction accuracy rate of an extraction system, many information-extraction systems at this stage are based on knowledge engineering.

Pinkston et al. [9] described a system in which an ontology was defined to extract attack information. This work was based on the revision of over 4000 intrusions and the strategies followed by them to attack a system; it introduces the means and consequences classes. Pinkston and coworkers adopted the rule-based approach by using the best practices of the semantic web. Rehman and Mustafa [10] described a system to extract information from a text description of vulnerability. In this system, they used a basic term frequency-inverse document frequency (TFIDF) score to extract information from the common vulnerabilities and exposures (CVEs). Lowis and Accorsi [11] presented another approach to classify service-oriented architecture (SOA) vulnerabilities: They used simple string matching on CVEs to classify datasets into various categories, and did not use machine learning techniques to classify CVEs. Their work is mainly focused on SOA vulnerabilities. The rule-based method has the advantage of high accuracy; however, for terms in which no obvious rules can be found, another method must be used for extraction.

The second main method is based on machine learning. The basic step involves training an information-extraction model using a large quantity of training data; the information-extraction model can then be used to extract related information. This method does not require rules defined by professionals in advance, although it requires a sufficient amount of training data to achieve good results.

Lal [12] proposed a system that can identify relevant entities from unstructured text. This system mainly addresses cyberattacks and software vulnerabilities; Lal also trained an NER to identify entities related to cybersecurity. Mulwad et al. [13] developed a framework that is used to detect and extract information about vulnerabilities and attacks from web text. They trained a support vector machine (SVM) classifier to identify potential vulnerability descriptions. This classifier uses the standard unigram bag-of-words vector model. Once a potential vulnerability description is identified, the framework extracts security-related entities and concepts using standard named entity recognition tools such as Open Calais. The two works described the machine learning method to automatically extract security-related information from unstructured text. This method cannot identify security-related entities accurately until sufficient training data is available.

2.3. Cybersecurity knowledge bases

(1) Vulnerability database. The existing abundant vulnerability databases are the China National Vulnerability Database of Information Security (CNNVD) [14] and the US NVD [15], which contain a variety of vulnerabilities. Information on vulnerabilities includes vulnerability names, vulnerability descriptions, vulnerability priorities, damage methods, corresponding characteristics, and more. At present, the vulnerability databases established by China and the United States both follow a common naming standard, making vulnerabilities from completely different databases available in the same standard. This standard greatly facilitates the sharing of vulnerability information.

(2) Attack rule base. This knowledge base collects information on existing attacks. The information includes the attack name, attack type, protocol, attack characteristic, attack description, severity, and other properties. The Snort attack rule base is a relatively perfect attack rule base in which every rule is stored as a line in a file.

2.4. Knowledge-based reasoning

Knowledge-based reasoning can be roughly divided into symbol-based reasoning and statistical-based reasoning. In artificial intelligence research, symbol-based reasoning is generally based on classical logic (first-order predicate logic or propositional logic) or on classical logic variation (e.g., default logic). Symbol-based reasoning can not only infer a new relationship between entities from an existing knowledge map using rules, but also perform logical conflict detection on a knowledge graph. Statistical-based reasoning methods generally make use of the relationship machine learning methods. The new entity relationship is learned from the knowledge graph by statistical rules.

The purpose of type reasoning on knowledge maps is to learn the relationship between instances and concepts in a knowledge map. The SDType approach [16] uses the statistical distribution of attributes connected by a triplet or predicate to predict the type of an instance. This method can be used in any single data source knowledge map, but it cannot do type reasoning across datasets. Both Tipalo [17], a tool for automatically typing DBpedia entities, and the Linked Hypernyms Dataset (LHD) [18] use unique abstract data to extract instance types by specific patterns. Such methods rely on structured text data and cannot be extended to other repositories.

The schema induction method can be mainly subdivided into the inductive logic program (ILP)-based method and the association rules mining (ARM)-based method. The ILP-based method combines machine learning and logic programming techniques to enable the user to draw logical conclusions from both instance and background knowledge. In Ref. [19], Lehmann et al. proposed a method of defining axioms using the concept of a downward-refined operator to learn description logic, starting with the most general concept (top concept), and using a heuristic search to make the concept continually specialized to reach a definition of the concept. This approach was further extended in Ref. [20] to handle large-scale semantic data such as DBpedia. These methods are all implemented in DL-Learner [21]. Völker and Niepert [22] introduced a statistical method for generating conceptual relationships from knowledge maps, which access information through SPARQL queries to build transactional tables. It then uses the ARM-based method from the transaction table to extract some of the associated conceptual relationships. In their follow-up work, Fleischhacker and Völker [23] used negative-association rule-extracting techniques to study the concept of non-conceptual relations, and rich experimental results are given in Ref. [24].

3. Framework design

Fig. 1 describes the approach to build a cybersecurity knowledge base. The framework mainly involves three parts: a data source, the construction of the ontology and extraction of information related to cybersecurity, and the generation of a cybersecurity knowledge graph.

Data sources can be divided into structured data and unstructured data. This paper presents an information-extraction method that consists of the rule-based method and machine learning to extract cybersecurity-related entities. The ontology, which is shown in the bottom of the framework, lays a foundation for information extraction. Therefore, constructing a proper ontology is quite important.

As shown in Fig. 1, the knowledge is stored in the form of a graph. The knowledge graph is a concept that was first proposed by Google in 2012 [25]. It is a semantic network that stores entities and relations between entities in the form of a graph. The advantage of a knowledge graph is obvious: The efficiency of its associated queries is higher than that of traditional storage methods. It is a flexible storage form that is quite easy to update. The construction of vertical knowledge, which is used in cybersecurity, must consider the depth of knowledge and the overall hierarchical structure. Therefore, this work adopted a top-down approach and

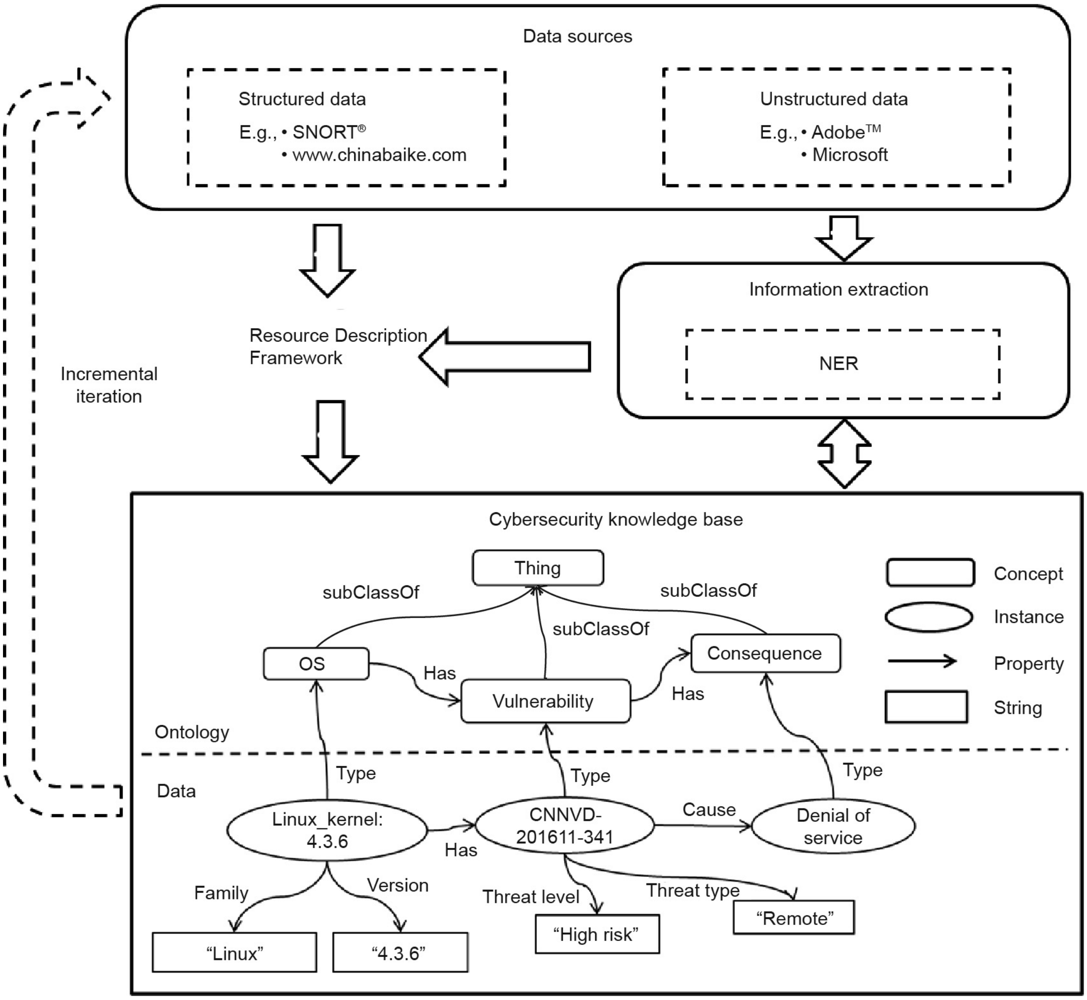

Fig. 1. Framework for constructing a cybersecurity knowledge graph. OS: operating system.

constructed the cybersecurity ontology first. Based on the ontology, cybersecurity information is extracted from structured and unstructured data. The following discussion focuses on ontology construction and knowledge extraction.

3.1. Construction of cybersecurity ontology

This paper proposes a quintuple cybersecurity knowledge base model comprising the aspects of concept, instance, relation, properties, and rule. The architecture of the cybersecurity knowledge base is shown in Fig. 2.

The architecture shown in Fig. 2 contains three ontologies: assets, vulnerability, and attack. Fig. 3 presents the cybersecurity ontology.

As shown in Fig. 3, there are five entity types, as follows:

- **Vulnerability**. Each of the records in the vulnerability database corresponds to an instance of a vulnerability type. Every vulnerability has its own unique CVE ID.
- **Assets**. The assets include the software and the operating system (OS) in this paper.
- **Software**. This is a subclass of assets (e.g., Adobe Reader).
- **OS**. This is a subclass of assets (e.g., Ubuntu 14.04).
- **Attack**. Most attacks can be regarded as an intrusion aimed at a certain vulnerability. The process of an attack can be a process of vulnerability exploitation.

3.2. Extraction of cybersecurity-related entities: A method based on machine learning

A conditional random field (CRF) is an undirected graph model based on statistical sequence identification and segmentation. The main idea of the model comes from the maximum entropy model. Its simplest form is the linear CRF, in which the nodes in the model form a linear structure. A linear CRF corresponds to a finite state machine, which is well suited for labeling linear data sequences.

NER problems can be defined as annotations of sequences—that is, in terms of whether or not the observed words belong to a predefined set of features. The CRF is the probability model of the annotation sequence. There is no independent assumption; therefore, features can be arbitrarily selected and globally normalized, and the global optimal solution can be obtained. It preserves the advantages of the conditional probability framework, such as the maximum entropy Markov model, and solves the problem of marker bias. Therefore, the CRF model is suitable for named entity recognition. Simple linear CRF is currently the best method for named entity recognition [26]. It models the probability distribution $P(y|x)$, in which x is the sequence of observation and y is the sequence of labels. $P(y|x)$ is calculated by the following formula:

$$P(y|x) \propto \prod_{j=1}^{N} \exp \sum_{i=0}^{M} \lambda_i f_i(y_{j-1}, y_j, x_j) \tag{1}$$

where N is the number of tokens and M is the number of features. Typically, f_i is binary and is shown as the following formula:

$$f_i(y_{j-1}, y_j, x_j) = \begin{cases} 1, & \text{if } y_{j-1} \text{ is OS, } y_j \text{ is OS, and } x_j \text{ is XP} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The Stanford NER [27] provides a general implementation of linear CRF sequence models. Therefore, it is also known as the CRFClassifier. In this study, we relied on the Stanford NER to extract cybersecurity-related entities. There are many options for features in the Stanford NER. We used the Stanford NER base implementation to train an extracting model, simply because our goal was also to train an NER in the cybersecurity domain.

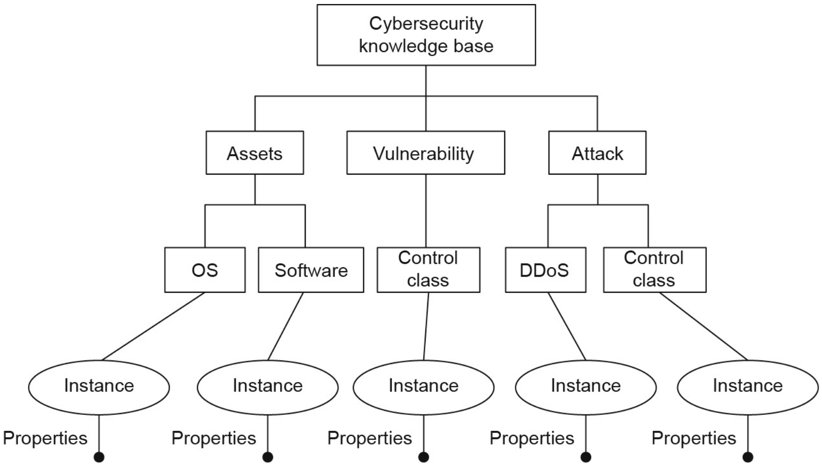

Fig. 2. The architecture of the cybersecurity knowledge base. DDoS: distributed denial of service.

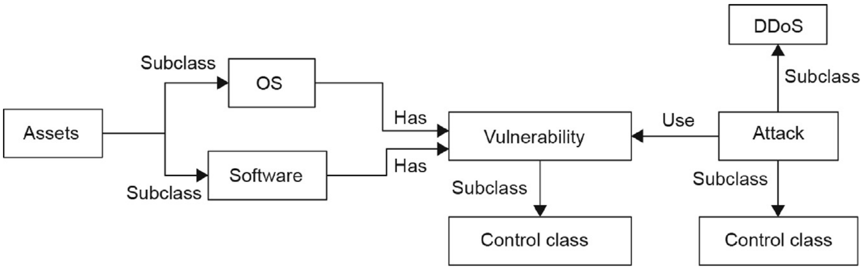

Fig. 3. The cybersecurity ontology.

It is important to choose features when building a model; here, a feature should be chosen that can better identify cybersecurity-related entities. A good combination of features is key in training a good extraction model. The Stanford NER provides over 70 features [28] that can be used to train a model. Determining proper features is not an easy task, because not much documentation exists about these features. Thus, the existing feature-selection algorithm was not helpful for our work. We had to analyze the features and select what we thought would be useful to train a model. We then verified our ideas by means of experiments. After many experiments, we determined a feature set that was used to train an NER and that achieved a good recognition effect. The features within the feature set we determined and used to train the NER are as follows:

- UseNGrams. This makes features from letter *n*-grams, that is, substrings of the word.
- MaxNGramLeng. The value type of this feature is "int." If the value of this feature is positive, *n*-grams above this value will not be used in the model. In this paper, we set the value of maxNGramLeng to 6.
- UsePrev. This can provide the feature for <previous word, class of previous word>; together with other options, it enables other previous features, such as <previous tag, class>. This results in features that are based on relationships between a current word and a pair: <previous word, class of previous word>. When there are successive words that belong to same class, this feature is very useful.
- UseNext. This is similar to the feature of UsePrev.
- UseWordPairs. This is based on two pairs: <previous word, current word, class> and <current word, next word, class>.
- UseTaggySequences. This is an important feature that uses the sequence of classes instead of words. It uses first-, second-, and third-order class and tag sequence interaction features.
- UseGazettes. If it is true, the next feature named gazette will point out the files by acting as an entity dictionary.
- Gazette. This value can be one or more filenames (i.e., names separated by a comma, semicolon, or space); if it is provided, entity dictionaries are loaded from these files. Each line should be an entity class name, followed by white space, and then followed by an entity (which might be a phrase of several tokens with a single space between words).
- CleanGazette. If true, this feature only fires when the whole word or phrase is matched in the gazette. For example, if the phrase "Windows 7" is present in the gazette, then the whole phrase should be matched in the document.
- SloppyGazette. If true, a gazette feature fires when any token of a gazette entry matches. In this example, "Windows" can be matched with "Windows 7."

This thesis uses the gazette feature for the software and OS classes. The Stanford NER provides two options to implement the gazette feature. Experiments proved the gazette feature and the cleanGazette option to be very good choices, as they improved recognition accuracy for the software and OS classes.

In order to use this feature, we summarized information from the field-of-influence platform in the vulnerability database and built an entity dictionary. The first column in the dictionary is the entity type, and the second column is the specific entity.

4. Knowledge deduction

4.1. Data source

Vulnerability includes existing vulnerabilities that can be collected. Attacks include the most popular attacking techniques today. The sources of the vulnerabilities are the CVE, the NVD, SecurityFocus, CXSECURITY, Secunia, the China National Vulnerability Database (CNVD), the CNNVD, and the Security Content Automation Protocol Chinese Community (SCAP). The data sources of the attack mainly include two categories: One category is from the information security website, which includes Pediy BBS, Freebuf, Kafan BBS, and the Open Web Application Security Project (OWASP). The other category is from the enterprise's self-built information-response center, including the 360 Security Response Center (360SRC) and the Alibaba Security Response Center (ASRC).

4.2. Principle analysis

K is used to represent the knowledge group; here, *K* = <concept, instance, relation, properties, rule>, where:

- Concept = {concept$_i$, i = 1, …, n}. The concept is a set of the abstract ontology, such as the OS, software, attack, and so on.
- Instance = {instance$_i$, i = 1, …, m}. The instance is a set of concrete examples, such as Windows 7, Adobe Reader, distributed denial of service (DDoS), and so on.
- Properties = {<instance$_i$, properties$_{ij}$, value$_j$>}. The properties are a set of instance attribute values.
- Relation = <concept$_i$, relation$_{cc}$, concept$_j$>|<concept$_i$, relation$_{ci}$, instance$_j$>|<instance$_i$, relation$_{ii}$, instance$_j$>. The relation shows the relationship between instances, such as subClassOf, instanceOf, is a (ISA), and so on.
- Rule = {rule|rule = <instance$_i$, new relation$_{ij}$, instance$_j$>|<concept$_i$, new relation$_{ij}$, instance$_j$>|<instance$_i$, properties$_{ij}$, new value$_j$>, based on *K*}. The rules are used to deduce new attribute values and new relationships.

The most important part of the knowledge group models presented in this paper is the rules. These rules can be used to deduce new relationships and new attribute values. The following discussion is devoted to how to deduce new attribute values and new relationships.

Attribute deduction uses the existing attributes of the instance and conditions to deduce new attributes. For example, one of Ming Yao's properties is his birthday. Given the current year and Ming Yao's birthday, his age can be obtained. As another example, if the record shows the number of times a host is scanned, then the total scanned number can be obtained using simple statistics.

Relationship deduction uses the relationships among existing instances to deduce the new relationships among the instances. For example, given Relation 1 (Si Li, birthplace, Beijing), Relation 2 (Si Li, residence, Shanghai), Relation 3 (Beijing, belongs to the country, China), Relation 4 (Shanghai, belongs to the country, China), Relation 5 (Si Li, nationality, Chinese), Relation 6 (Si Li, classmates, San Zhang), and Relation 7 (San Zhang, birthplace, Beijing), a new relationship can be obtained: Relation 8 (San Zhang, nationality, Chinese).

4.3. The result of the deduction

4.3.1. Attribute deduction

As shown in Fig. 4, there are three instances: N_i, N_j, and N_l. Each instance corresponds to keys and values.

Attributes are represented by (node, key, value) pairs. The prediction formula of the attribute value, Value$_{ik}$, is as follows:

$$\text{Value}_{ik} = \sum_{j=1}^{m} \lambda_j \cdot f_{ij}(\text{key}_j, \text{value}_j) + \sum_{t=1}^{l} \sigma_t \cdot \sum_{j=1}^{m} \lambda_j \cdot f_{ij}(\text{key}_j + \text{value}_j) \tag{3}$$

For N_i, the new attribute can be obtained by calculating the above formula, as shown in Fig. 5.

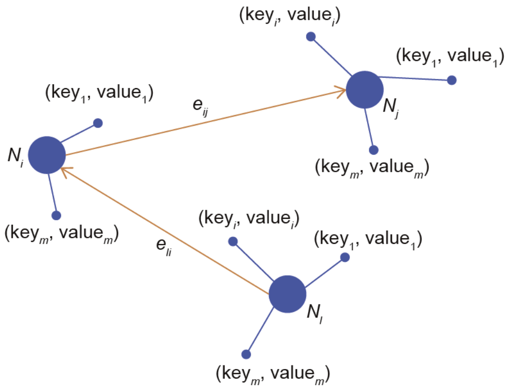

Fig. 4. Instances of the attribute.

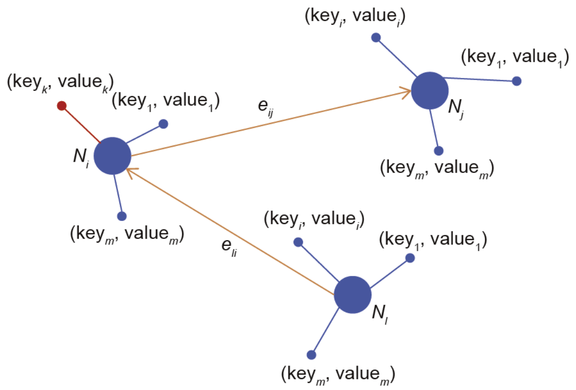

Fig. 5. Generating a new attribute value. The red part of N_i is the new attribute value.

4.3.2. Relationship deduction

There are three types of commonly used reasoning methods: embedding-based technology, which is based on low-dimensional vector representation; path-ranking algorithms, which are traditional path-sorting algorithms; and probabilistic graphical models such as the Markov chain. In this paper, we use the path-sorting algorithms. The basic idea of path sorting is to use the path connecting two entities as a feature to predict the relationship between the two entities. Using the path-sorting algorithms for a given relationship, we can determine whether such a relationship exists between the two entities.

The attribute values and relationships among the instances N_i, N_j, and N_l are shown in Fig. 6:

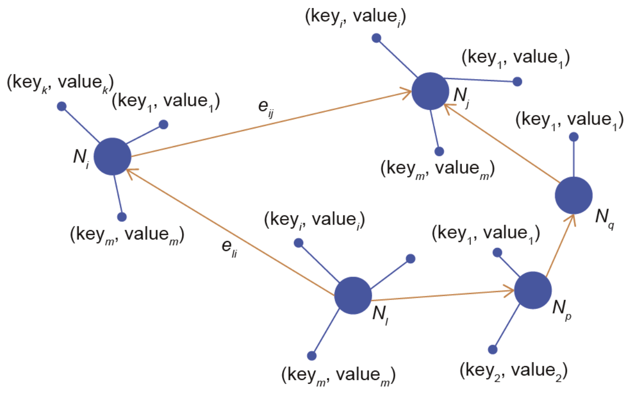

Fig. 6. Relationships between instances.

The predictive formula of relational reasoning is as follows:

$$\text{Score}(l,j) = \sum_{\pi \in Q} \text{Path}[e_l, e_j; \text{length}(\pi) \leqslant n] \cdot \omega_\pi \tag{4}$$

where ω_π is the weight of the reachable path, π, from i to j. A length $\pi \le n$ indicates that the path length is less than or equal to n. In this case, if $\text{Score}(l,j) \geqslant \tau$, τ is the threshold, then e_{lj} is established; otherwise, e_{lj} is invalid.

Through the path-sorting algorithm, a new relationship can be obtained, as shown in Fig. 7.

4.4. Evaluation criteria

In information retrieval and extraction systems, there are two main evaluation indicators: precision and recall. Sometimes, in order to comprehensively evaluate the performance of the system, the harmonic mean of precision and recall is calculated. This is usually referred to as the *F*-measure. In this paper, we used F_1, which is a special form of the *F*-measure. Precision, recall, and F_1 are defined by true positives, false positives, and false negatives. The definition is as follows:

- True positives (TP): This is a collection of the class members that are correctly labeled as belonging to a particular class.
- False positives (FP): This is a collection of the class members that are mislabeled as belonging to a particular class.
- False negatives (FN): This is a collection of the items that are not labeled as any class by the system, although they actually belong to some class.

Precision is given by:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

Recall is given by:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

F_1 is given by:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

4.5. Experimental results

We merged the annotated data from four data sources into one dataset. To verify the influence of useGazettes, we built three models (NER1, NER2, and NER3). NER1 did not use useGazettes as its feature, while NER2 used useGazettes and chose the option of cleanGazette. NER3 also used useGazettes, but its option was

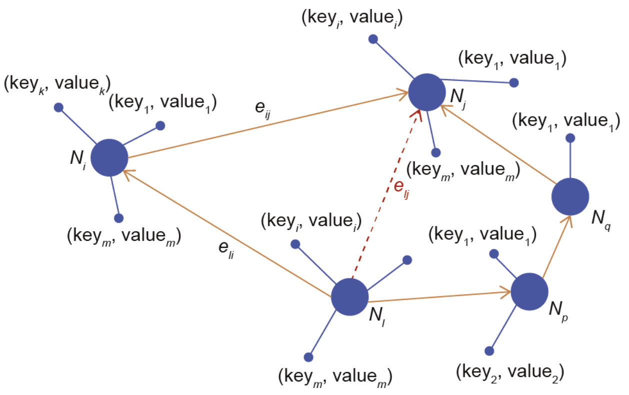

Fig. 7. A new relationship between instances. The red dotted line with the arrow is the new relationship, e_{lj}.

sloppyGazette. We then adopted an approach of 10-fold cross-validation to evaluate these models. We divided the data into 10 data blocks. Nine tenths of the blocks were used as training data, and the rest was used as testing data. The following tables show the average of the precision, recall, and F_1 measures for the three models.

At the beginning, we chose features without useGazettes to train the NER1. The average recognition results of NER1 are shown in Table 1; the table shows that the precision of consequence is relatively high. Regarding the F_1 measure, the recognition rates of software and vulnerability are close to each other and are higher than those of any other entity type; that is, the recognition of software and vulnerability achieved good performance in terms of overall recognition effect.

Table 1
Recognition results of NER1.

Entity	Precision	Recall	F_1
Software	0.700	0.795	0.745
OS	0.779	0.691	0.732
Vulnerability	0.805	0.689	0.743
Attack	0.822	0.597	0.692
Total	0.739	0.735	0.737

We then used features that included useGazettes and chose the option of cleanGazette to train NER2. NER3 was trained based on features that included useGazettes with sloppyGazette as its option. The average recognition results are shown in Table 2.

As shown in Table 2, in the recognition results of NER2, the recognition for software achieved good overall performance. For NER3, the recognition for OS achieved a high F_1 value. In terms of the recognition of software and OS, NER3 achieved better overall performance than NER2. This result indicates that the sloppyGazette option is helpful for recognizing cybersecurity-related entities. The F_1 measures of both consequence and mean in NER2 and NER3 are still low, both being less than 70%. An intuitive comparison between the three models is presented in Fig. 8.

As shown in Fig. 8, in terms of the recognition accuracy of software and OS, including the recall and F_1 measure, both NER2 and NER3 are higher than NER1. This result verified the importance of useGazettes in training an NER in the cybersecurity domain. Regarding consequence and mean, an entity dictionary is difficult to build. Without gazette, the recognition accuracy was not high.

Table 2
Recognition results of NER2 and NER3.

Model	Entity	Precision	Recall	F_1
NER2 (cleanGazette)	Software	0.809	0.838	0.823
	OS	0.752	0.875	0.809
	Vulnerability	0.753	0.632	0.688
	Attack	0.884	0.559	0.685
	Total	0.789	0.799	0.794
NER3 (sloppyGazette)	Software	0.877	0.838	0.857
	OS	0.832	0.904	0.866
	Vulnerability	0.775	0.632	0.696
	Attack	0.875	0.538	0.667
	Total	0.852	0.805	0.828

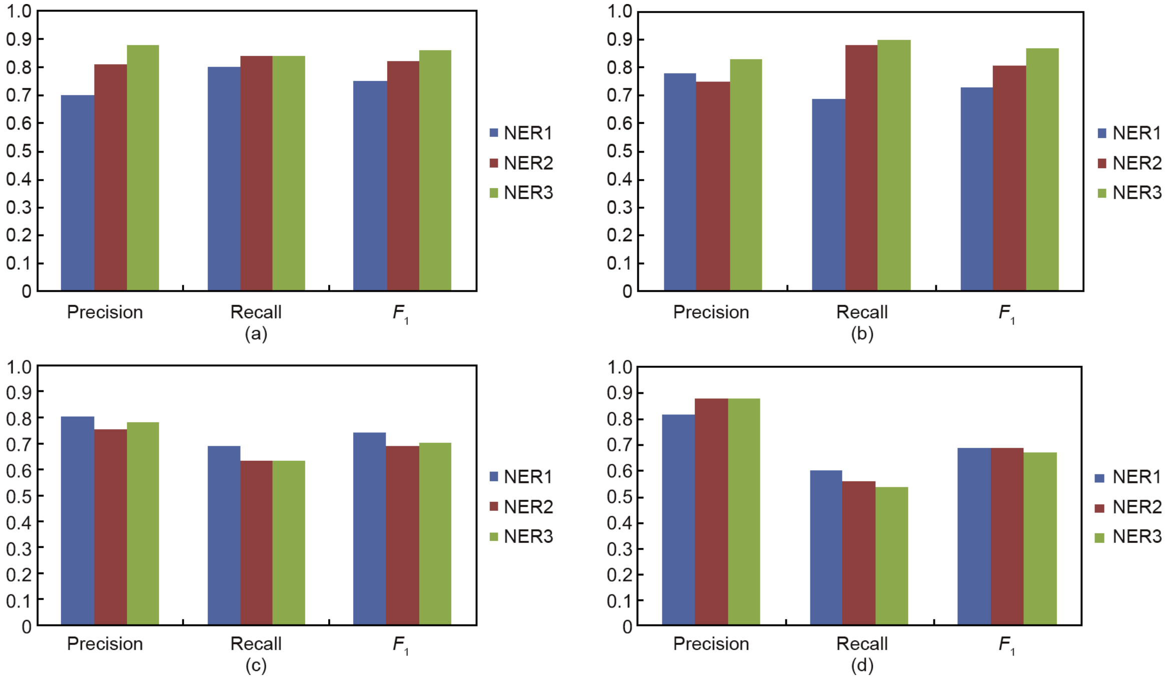

Fig. 8. Recognition results for each entity type. (a) Software; (b) OS; (c) consequence; (d) mean.

5. Conclusion and future work

This paper builds an ontology for cybersecurity that is based on vulnerability, and puts forward a method to build a cybersecurity knowledge base. The Stanford NER was used to train an extractor to extract cybersecurity-related entities; however, the recognition accuracy needs further improvement. In addition, new entity attributes and new relationships between entities can be obtained using deductive rules.

In future, the most important work will be to enrich the cybersecurity knowledge base and the rules of deduction, and then apply them to the research of intrusion detection and situational awareness.

Acknowledgements

We are grateful for the support of the National Natural Science Foundation of China (U163215, 61472433, 61732022, 61732004, 61672020, and 61502517) and the National Key Research and Development Program (2016YFB0800802, 2016YFB0800803, 2016YFB0800804, 2017YFB0802204, 2016QY03D0601, 2016QY03D0603, and 2016YFB0800303).

Compliance with ethics guidelines

Yan Jia, Yulu Qi, Huaijun Shang, Rong Jiang, and Aiping Li declare that they have no conflict of interest or financial conflicts to disclose.

References

[1] Zhu J, Zhang J, Zhang C, Wu Q, Jia Y, Zhou B, et al. CHRS: Cold start recommendation across multiple heterogeneous information networks. IEEE Access 2017;5:15283–99.

[2] Zhu X, Huang J, Zhou B, Li A, Jia Y. Real-time personalized twitter search based on semantic expansion and quality model. Neurocomputing 2017;254:13–21.

[3] Undercoffer J, Joshi A, Pinkston J. Modeling computer attacks: An ontology for intrusion detection. In: Vigna G, Jonsson E, Kruegel C, editors. RAID 2003: Recent advances in intrusion detection. Berlin: Springer; 2003. p. 113–35.

[4] Joshi A, Lal R, Finin T, Joshi A. Extracting cybersecurity related linked data from text. In: Proceedings of the 7th IEEE international conference on semantic computing. Los Alamitos: IEEE Computer Society Press; 2013. p. 252–9.

[5] More S, Matthews M, Joshi A, Finin T. A knowledge-based approach to intrusion detection modeling. In: Proceedings of 2012 IEEE symposium on security and privacy workshops. Los Alamitos: IEEE Computer Society Press; 2012. p. 75–81.

[6] Obrst L, Chase P, Markeloff R. Developing an ontology of the cybersecurity domain. CEUR Workshop Proc 2012;966:49–56.

[7] Parmelee MC. Toward an ontology architecture for cyber-security standards. CEUR Workshop Proc 2010;713:116–23.

[8] Iannacone M, Bohn S, Nakamura G, Gerth J, Huffer K, Bridges R, et al. Developing an ontology for cybersecurity knowledge graphs. In: Proceedings of the 10th annual cyber and information security research conference. New York: ACM, Inc.; 2015.

[9] Pinkston J, Undercoffer J, Joshi A, Finin T. A target-centric ontology for intrusion detection. In: Proceedings of the IJCAI-03 workshop on ontologies and distributed systems, Aug 9–15, 2003, Acapulco, Mexico; 2003. p. 47–58.

[10] Rehman S, Mustafa K. Software design level vulnerability classification model. Int J Comput Sci Secur 2012;6(4):238–55.

[11] Lowis L, Accorsi R. On a classification approach for SOA vulnerabilities. In: Proceedings of the 33rd annual IEEE international computer software and applications conference. Los Alamitos: IEEE Computer Society Press; 2009. p. 439–44.

[12] Lal R. Information extraction of cybersecurity related terms and concepts from unstructured text [dissertation]. College Park: University of Maryland; 2013.

[13] Mulwad V, Li W, Joshi A, Finin T, Viswanathan K. Extracting information about security vulnerabilities from web text. In: Hübner JF, Petit JM, Suzuki E, editors. Proceedings of 2011 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology—workshops. Los Alamitos: IEEE Computer Society Press; 2011. p. 257–60.

[14] CNNVD.org.cn [Internet]. Beijing: China Information Technology Security Evaluation Center; [cited 2017 Jul 25]. Available from: http://www.cnnvd.org.cn/. Chinese.

[15] NVD.nist.gov [Internet]. Gaithersburg: National Institute of Standards and Technology; [cited 2017 Jul 25]. Available from: https://nvd.nist.gov/.

[16] Paulheim H, Bizer C. Type inference on noisy RDF data. In: Alani H, Kagal L, Fokoue A, Groth P, Biemann C, Parreira JX, et al., editors. The semantic web—ISWC 2013: Proceedings of the 12th international semantic web conference. Berlin: Springer; 2013. p. 510–25.

[17] Paulheim H, Bizer C. Type inference on noisy RDF data. In: Cudré-Mauroux P, Heflin J, Sirin E, Tudorache T, Euzenat J, Hauswirth M, et al., editors. The semantic web—ISWC 2012: Proceedings of the 11th international semantic web conference. Berlin: Springer; 2012. p. 65–81.

[18] Kliegr T. Linked hypernyms: Enriching DBpedia with targeted hypernym discovery. J Web Semant 2015;31:59–69.

[19] Lehmann J, Auer S, Bühmann L, Tramp S. Class expression learning for ontology engineering. J Web Semant 2011;9(1):71–81.

[20] Hellmann S, Lehmann J, Auer S. Learning of OWL class descriptions on very large knowledge bases. Int J Semant Web Inf Syst 2009;5(2):25–48.

[21] Lehmann J. DL-learner: Learning concepts in description logics. J Mach Learn Res 2009;10(11):2639–42.

[22] Völker J, Niepert M. Statistical schema induction. In: Antoniou G, Grobelnik M, Simperl E, Parsia B, Plexousakis D, De Leenheer P, et al., editors. The semantic web: Research and applications: Proceedings of the 8th extended semantic web conference. Berlin: Springer; 2011. p. 124–38.

[23] Fleischhacker D, Völker J. Inductive learning of disjointness axioms. In: Meersman R, Dillon T, Herrero P, Kumar A, Reichert M, Qing L, et al., editors. On the move to meaningful internet systems: OTM 2011: Proceedings of confederated international conferences: CoopIS, DOA-SVI, and ODBASE 2011. Berlin: Springer; 2011. p. 680–97.

[24] Völker J, Fleischhacker D, Stuckenschmidt H. Automatic acquisition of class disjointness. J Web Semant 2015;35(Pt 2):124–39.

[25] Singhal A. Introducing the knowledge graph: Things, not strings [Internet]. [updated 2012 May 16; cited 2017 Jul 25]. Available from: https://googleblog.blogspot.com/2012/05/introducing-knowledge-graphthings-not.html.

[26] Lin D, Wu X. Phrase clustering for discriminative learning. In: Proceedings of the 47th annual meeting of the association for computational linguistics and the 4th international joint conference on natural language processing of the AFNLP. Singapore: Suntec; 2009. p. 1030–8.

[27] Finkel JR, Grenager T, Manning C. Incorporating non-local information into information extraction systems by Gibbs sampling. In: Knight K, Ng HT, Oflazer K, editors. Proceedings of the 43rd annual meeting of the association for computational linguistics. Stroudsburg: Association for Computational Linguistics; 2005. p. 363–70.

[28] NERFeatureFactory [Internet]. Stanford: Stanford NLP Group; [updated 2013 Jun 26; cited 2017 Jul 25]. Available from: http://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/ie/NERFeatureFactory.html.