A Practical Approach to Constructing a Knowledge Graph for Cybersecurity

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

Xiaofan Yan

Outline

- Abstract
- Introduction
- Related works
- Framework design
- Knowledge deduction
- Conclusion and future work

Abstract

- Cyberattack forms are complex and varied
  - detection and prediction of dynamic types of attack
- Presents a cybersecurity knowledge base and deduction rules based on a quintuple model.
  - Extract entities and build ontology
  - Calculating formulas and using the path-ranking algorithm
  - Stanford NER used to train an extractor to extract useful information
  - Stanford NER provides many features and the useGazettes parameter

Introduction

- Building a cybersecurity knowledgebase following a three-step procedure
  - First, obtain information -- collect/analyze structured/unstructured data
  - Second, construct the ontology according to the obtained information
  - Third, generate the cybersecurity knowledge base.
- Cybersecurity knowledge deduction
  - A quintuple model
  - Path-ranking algorithm

Related works

- Ontology construction
- Information extraction
  - based on knowledge engineering
  - based on machine learning
- Cybersecurity knowledge bases
  - Vulnerability database
  - Knowledge-based reasoning
- Knowledge-based reasoning
  - symbol-based reasoning
  - statistical-based reasoning

Framework design

- 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**.

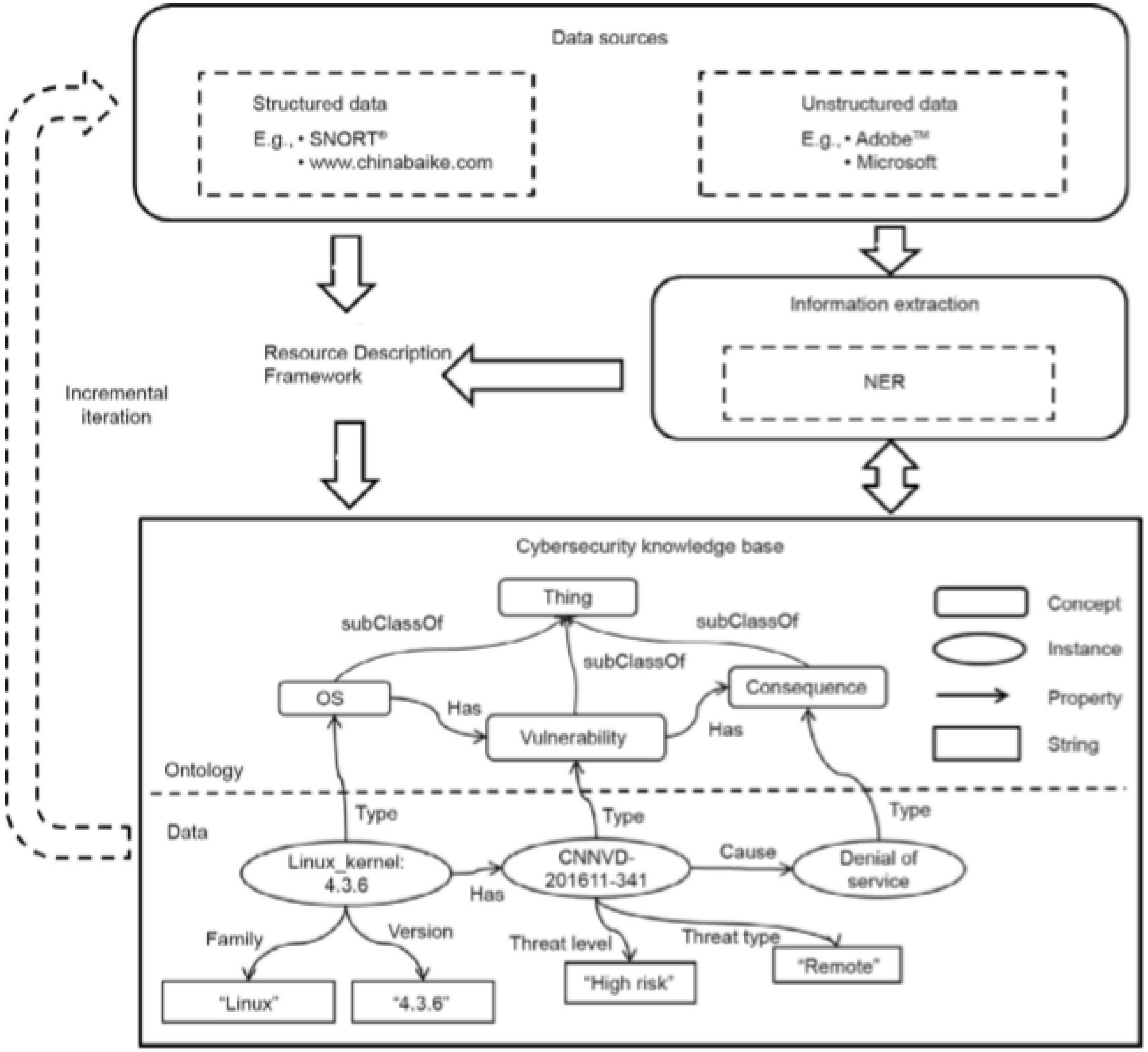

Fig. 1. Framework for constructing a cybersecurity knowledge graph. OS: operating system.

Framework design

- Construction of cybersecurity ontology
  - Three ontologies: assets, vulnerability, and attack
  - five entity types:
    - Vulnerability.
    - Assets.
    - Software.
    - OS.
    - Attack.

Framework design

- Extraction of cybersecurity-related entities: A method based on machine learning
  - CRF
    - Simple linear CRF is currently the best method for named entity recognition
    - Models the probability distribution P(y|x), in which x is the **sequence of observation** and y is the **sequence of labels**.
  - Relied on the Stanford NER to extract cybersecurity-related entities
    - Stanford NER base implementation to train an extracting model
    - The Stanford NER provides over 70 features
    - Determined a feature set
      - UseNGrams, MaxNGramLeng, UsePrev, UseNext, UseWordPairs, UseTaggySequences, UseGazettes, Gazette, CleanGazette, SloppyGazette

Knowledge deduction

- Data source
  - Vulnerability: the CVE, the NVD, SecurityFocus, CXSECURITY, Secunia, the China National Vulnerability Database (CNVD), the CNNVD, and the Security Content Automation Protocol Chinese Community (SCAP).
  - Attack:
    - From the information security website: Pediy BBS, Freebuf, Kafan BBS, and the Open Web Application Security Project (OWASP).
    - From the enterprise’s self-built information-response center: 360 Security Response Center (360SRC) and the Alibaba Security Response Center (ASRC)

Knowledge deduction

- Principle analysis
  - K is used to represent the knowledge group
    - K = <concept, instance, relation, properties, rule>
    - Concept = {concepti, i = 1, ... , n}.
    - Instance = {instancei, i = 1,... , m}.
    - Properties = {<instancei,propertiesij,valuej>}.
    - Relation = <concepti, relationcc, conceptj>|<concepti, relationci, instancej>|<instancei, relationii, instancej>.
    - Rule = {rule|rule = <instancei, new relationij, instancej>|<concepti, new relationij, instancej>|<instancei, propertiesij, new valuej>, based on K}.
    - These rules can be used to deduce new relationships and new attribute values.

Knowledge deduction

- The result of the deduction
  - Attribute deduction
  - Relationship deduction
  - Evaluation criteria
  - Experimental results
- Evaluation criteria
  - Precision and recall

Knowledge deduction

- Experimental results
  - NER1 did not use useGazettes as its feature
  - NER2 used useGazettes and chose the option of cleanGazette
  - NER3 also used useGazettes, but its option was sloppyGazette.

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

Knowledge deduction

- Experimental results
  - NER1 did not use useGazettes as its feature
  - NER2 used useGazettes and chose the option of cleanGazette
  - NER3 also used useGazettes, but its option was sloppyGazette.

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

Conclusion and future work

- Builds an ontology for cybersecurity that is based on vulnerability, and puts forward a method to build a cybersecurity knowledge base.