Suitable is the Best: Task-Oriented Knowledge Fusion in Vulnerability Detection

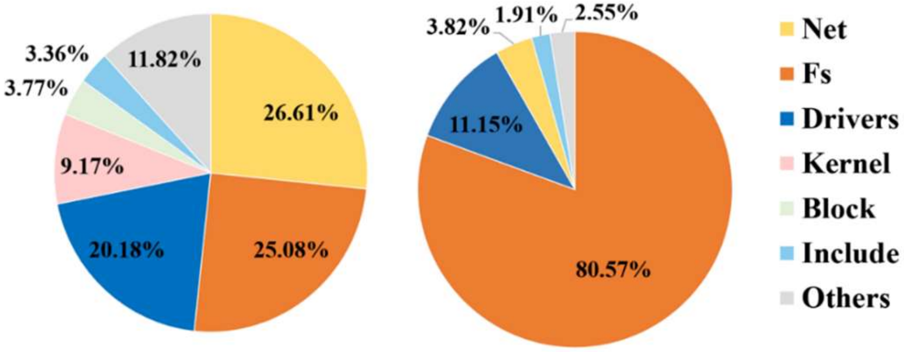
Jingjing Wang, Minhuan Huang, Yuanpin Nie, Xiang Li, Qianjin Du, Wei Kong, Huan Deng, Xiaohui Kuang

Academy of Military Sciences,
Tsinghua University,
Zhejiang Sci-Tech University

NeurIPS 2024

Motivation

- Examples



The distribution of CWE-416 (left) and CWE-119 (right) vulnerabilities across all modules in the Linux kernel over the past decade.

```
@@ -2134,9 +2135,11
@@ static int
netlink_dump(struct sock *sk)
{     ...
      nlk->cb_running = false;
+     module = cb->module;
+     skb = cb->skb;
      mutex_unlock(nlk->cb_mutex);
-     module_put(cb->module);
-     consume_skb(cb->skb);
+     module_put(module);
+     consume_skb(skb);
      return 0;
      ...
}
```

```
@@ -4255,9 +4258,8
@@static int
nft_set_desc_concat_parse(const
struct nlattr *attr,struct
nft_set_desc *desc)
{     ...
      len =
ntohl(nla_get_be32(tb[NFTA_SET_FIELD_
LEN]));

-     if (len * BITS_PER_BYTE / 32 >
            NFT_REG32_COUNT)
-           return -E2BIG;
+     if (!len || len > U8_MAX)
+           return -EINVAL;
      desc->field_len[desc
->field_count++] = len;
      ...
}
```

CWE-416 (left) and CWE-119 (right) discovered in the net module.

- Key observations

Potential **vulnerability patterns** associated with **program behavior** differ and have distinct characteristics depending on **the context of different detection targets and tasks**.

Motivation

- Problem

Existing deep learning-based vulnerability detection methods primarily employ a uniform and consistent feature learning pattern across the entire target :

General-Purpose Detection Tasks

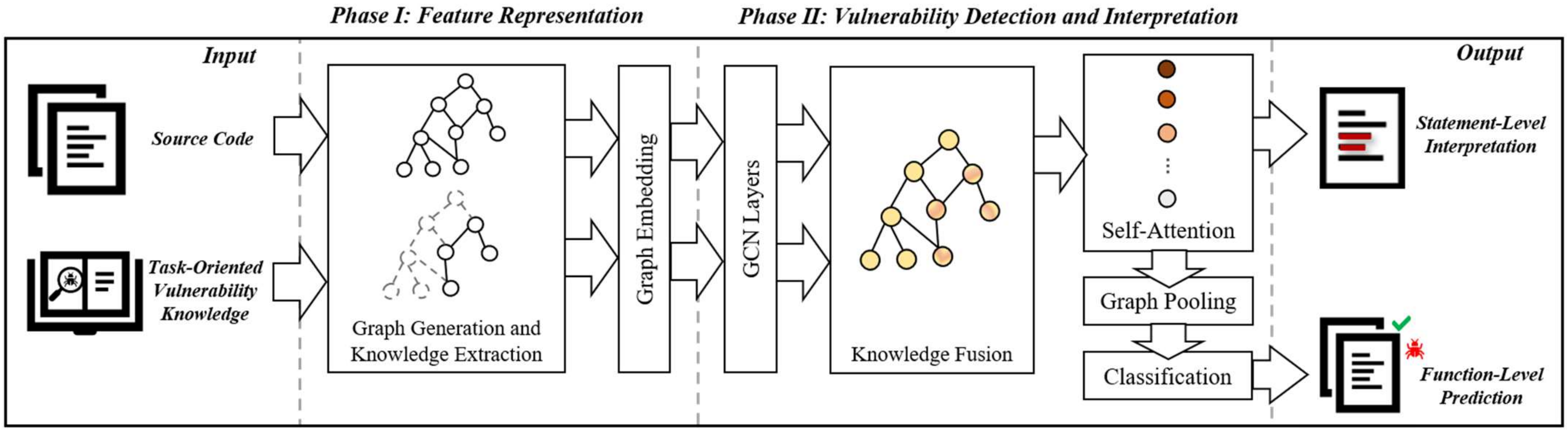
(1) *Focusing on* ***target code projects*** (without concern for specific vulnerability types) : IVDetect, Reveal…

(2) *Focusing on* ***specific vulnerability types*** (existing in different code projects): Vuldeepecker, Ubitect…

Difficult to make full use of known information in **diverse practical task scenarios** to **characterize the potential vulnerability characteristics** of different target codes.

The KF-GVD Framework

- The overall architecture of KF-GVD



KF-GVD, a **K**nowledge **F**usion-based **G**NN model for source code **V**ulnerability **D**etection.

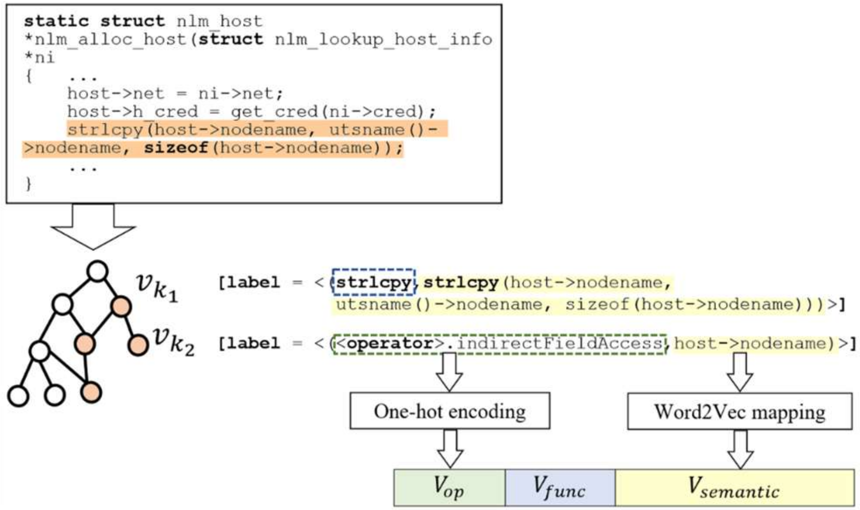
The KF-GVD Framework

- Feature representation
  - Code property graph generation
  - Task-oriented vulnerability knowledge extraction

    Vulnerable program operations

    Sensitive functions

    Customized knowledge for specific tasks



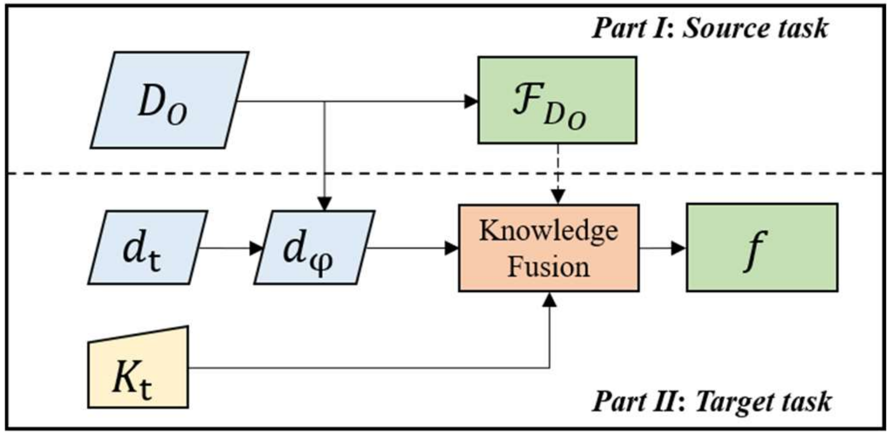
  - Graph embedding

    Node feature vectors

    Adjacency matrix

The KF-GVD Framework

- The Workflow of KF-GVD



The training of model f for a subtask t :

1) Dataset collection.
2) Initialize the parameters of f using F_{D_O}.
3) Perform feature fusion only on the data d_t' randomly sampled from d_t:

$$h_{u_j} = \begin{cases} Fusion(\alpha h_{v_j}, \beta h_{v_q}), v_j \in V_k \\ h_{v_j}, v_j \notin V_k \end{cases}$$

Evaluation

- Comparison of Function-Level Vulnerability Detection Results

Method	S_{416}			T_{m416} Net			Fs			Drivers			Kernel			Block			Include		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Cppcheck	27.7	42.6	33.6	14.8	22.7	17.9	27.0	53.6	35.9	30.7	45.9	36.8	10.3	45.9	16.8	30.2	36.5	33.1	23.6	31.8	27.1
Flawfinder	33.4	45.9	38.7	20.6	36.6	26.4	15.9	42.6	23.2	5.6	22.4	9.0	28.5	62.8	39.2	17.8	26.7	21.4	25.0	39.7	30.7
Sysver	58.4	67.2	62.5	21.9	40.5	28.4	27.2	37.3	31.5	32.5	30.7	31.6	22.7	23.6	23.1	37.9	30.2	33.6	26.3	45.7	33.4
VulCNN	66.9	72.8	69.7	33.4	52.7	40.9	47.0	52.4	49.6	28.5	43.1	34.3	36.8	56.3	44.5	24.7	65.1	35.8	22.5	39.6	28.7
Codebert	66.2	62.3	64.2	50.3	42.2	45.9	47.8	36.7	41.5	42.3	51.6	46.5	46.5	51.1	48.7	42.9	40.8	41.8	40.9	35.1	37.8
CodeLlama	65.9	59.1	62.3	52.9	46.0	49.2	50.6	52.6	51.6	44.1	43.9	44.0	53.3	51.5	52.4	40.5	41.1	40.8	57.6	52.8	55.1
Wizardcoder	59.6	69.3	64.1	53.7	48.2	50.8	42.7	38.2	40.3	39.8	52.0	45.1	55.6	50.5	52.9	44.3	38.5	41.2	56.4	49.3	52.6
Devign	63.7	79.4	70.7	37.1	42.6	39.7	48.9	50.2	49.5	34.1	56.9	42.6	37.5	44.6	40.7	48.1	33.9	39.8	36.8	70.4	48.3
ReGVD	67.2	71.7	69.4	41.9	43.7	42.8	50.3	51.5	50.9	40.6	45.9	43.1	45.8	55.5	50.2	42.8	34.8	38.4	44.5	65.2	52.9
IVDetect	81.8	**94.7**	87.8	40.4	36.2	38.2	51.7	51.9	51.8	43.8	41.2	42.4	41.4	60.0	49.0	39.0	35.6	37.2	66.7	80.0	72.7
GVD-ft	86.8	89.3	88.0	53.6	88.2	66.7	73.3	66.0	69.5	41.3	44.7	42.9	39.5	44.7	42.0	51.7	78.4	51.9	50.0	53.3	51.6
KF-GVD	86.8	89.3	88.0	78.6	98.1	87.3	73.9	94.4	82.9	87.1	71.8	78.7	85.4	92.1	88.6	66.7	83.3	74.1	82.4	93.3	87.5

Method	S_{119}			T_{m119} Fs			Drivers			Net			Include			T_{sub} CWE-125			CWE-787		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Cppcheck	45.0	55.7	49.8	33.7	50.5	40.4	32.1	45.9	37.8	44.2	40.0	42.0	23.9	35.7	28.6	24.8	50.6	33.3	29.4	35.7	32.2
Flawfinder	27.6	50.4	35.7	15.3	57.4	24.2	25.9	44.8	32.8	37.6	42.8	40.0	29.7	56.8	39.0	12.9	37.4	19.2	18.3	33.5	23.7
Sysver	54.8	70.6	61.7	23.6	67.2	34.9	28.3	56.2	37.6	15.7	60.9	25.0	33.0	42.6	37.2	39.7	58.4	47.3	33.4	48.6	39.6
VulCNN	63.9	77.4	70.0	35.5	50.7	41.8	27.8	44.6	34.3	39.4	58.6	47.1	22.0	43.5	29.2	16.8	29.1	21.3	17.6	33.0	23.0
Codebert	65.2	67.9	66.5	54.7	39.5	45.9	37.5	40.0	38.7	48.5	44.1	46.2	34.6	51.8	41.5	34.8	57.6	43.4	43.7	48.6	46.0
CodeLlama	70.0	64.1	66.9	55.7	54.9	55.3	45.6	45.8	45.7	57.2	48.0	52.2	49.3	53.9	51.5	37.6	53.9	44.3	48.0	55.8	51.6
Wizardcoder	72.4	52.4	60.8	62.5	35.5	45.3	45.8	48.7	47.2	50.8	42.0	46.0	48.6	56.6	52.3	33.5	52.5	40.9	47.5	51.9	49.6
Devign	68.5	70.2	69.3	30.6	54.2	39.1	35.4	42.8	38.7	48.6	57.2	52.6	25.8	40.3	31.5	20.1	37.9	26.3	18.4	25.0	21.2
ReGVD	74.1	71.2	72.6	60.8	34.2	43.8	40.9	47.1	43.8	52.1	59.1	55.4	44.1	50.8	47.2	29.8	54.0	38.4	44.9	57.2	50.3
IVDetect	79.0	83.3	81.1	46.7	33.3	38.9	33.3	66.7	44.4	66.7	50.0	57.1	40.0	46.2	42.9	31.9	55.8	38.1	46.8	52.4	43.0
GVD-ft	82.9	90.9	86.7	73.5	58.7	65.2	66.7	88.9	76.2	76.3	58.5	64.7	57.1	61.5	59.3	49.8	60.5	54.6	66.7	61.5	64.0
KF-GVD	82.9	90.9	86.7	96.1	95.2	95.7	90.0	94.7	92.3	91.7	75.0	82.5	91.7	84.6	88.0	59.2	80.0	67.9	80.0	84.2	82.1

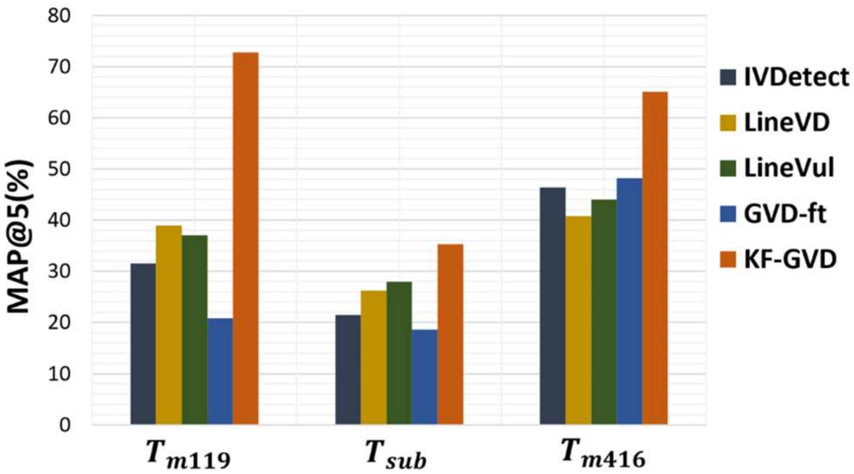
KF-GVD demonstrates an improvement in precision by 0.6%-44%, recall by 5.8%-29.3%, and an average gain of 22.6% on F1-score.

Evaluation

- Comparison of Statement-Level Vulnerability Detection Results

Method	T_{m119}												T_{sub}					
	Fs			Drivers			Net			Include			CWE-125			CWE-787		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
IVDetect	32.3	56.1	34.8	10.5	63.1	15.4	36.7	20.4	26.0	9.7	74.7	16.4	2.2	17.1	3.1	16.7	10.0	12.5
LineVD	39.2	27.9	32.6	11.0	58.7	16.1	37.6	21.2	26.8	17.2	53.2	26.1	4.1	24.9	5.3	**33.3**	20.0	25.0
LineVul	33.8	45.0	38.6	10.7	24.0	14.8	22.4	28.0	24.9	16.3	44.8	23.9	6.4	13.6	8.7	29.8	19.0	23.2
GVD-ft	32.1	55.0	34.5	11.2	66.0	16.4	18.3	10.2	13.0	9.6	**85.4**	16.3	7.5	51.0	10.3	2.9	1.8	2.2
KF-GVD	82.1	58.7	66.6	38.2	81.1	49.6	74.7	65.5	66.3	54.9	84.4	65.0	31.9	55.8	38.1	29.2	67.9	31.4

Method	T_{m416}																	
	Net			Fs			Drivers			Kernel			Block			Include		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
IVDetect	19.6	58.1	24.5	15.4	**80.8**	19.2	20.2	77.9	25.5	27.7	**83.8**	36.9	15.4	23.6	18.6	67.9	67.7	67.5
LineVD	24.0	**98.8**	31.3	16.6	55.9	25.6	17.9	75.2	23.3	15.8	72.9	21.9	12.5	16.7	14.3	48.2	49.2	48.7
LineVul	20.9	45.3	28.6	15.3	44.0	22.7	22.8	32.8	26.9	24.9	41.8	31.2	14.1	48.0	21.8	31.7	36.4	33.9
GVD-ft	22.7	58.6	25.3	16.7	71.3	21.8	25.3	69.9	28.0	16.4	66.5	22.4	10.8	55.3	15.2	52.9	52.2	52.4
KF-GVD	56.3	96.3	63.8	55.9	80.8	66.0	76.5	81.1	68.1	80.6	75.9	75.1	27.4	97.3	36.1	73.3	73.1	72.6



KF-GVD achieves an average improvement of 59.7% in precision, 30.9% in recall, and 42.4% in MAP@5.

Case Study

- Undisclosed Vulnerabilities Detected by KF-GVD

ID	Project	File Location	Vul_line
CNNVD-2023-43767151	assimp	/…/OpenDDLParser.cpp	348
CNNVD-2023-12599427		/…/FBXParser.cpp	192
CNNVD-2023-59936877	boost	/…/detail/rapidxml.hpp	644
CNNVD-2023-23489133		/…/basic_regex_creator.hpp	710
CNNVD-2023-20301510	c-blosc2	/…/blosc-private.h	120
CNNVD-2023-76730942	exiv2	/…/value.cpp	13
CNNVD-2023-90736138	flatbuffers	/…/util.h	133
CNNVD-2023-83881569	frr	/…/bgp_attr.c	2658
CNNVD-2023-27702356	harfbuzz	/…/hb-atomic.hh	172

Suitable is the Best: Task-Oriented Knowledge Fusion in Vulnerability Detection

Jingjing Wang, Minhuan Huang, Yuanpin Nie, Xiang Li,
Qianjin Du, Wei Kong, Huan Deng, Xiaohui Kuang

- Jingjing Wang: jennywangel@163.com
- Supplementary Material: https://github.com/fgVDgnn/KF-GVD/tree/master

NeurIPS 2024