APPENDIX A
APPENDIX: EXAMPLES OF OUR PROMPT TEMPLATES

A. Legend

- Gray text represents sections of text which are excluded for brevity or conciseness.
- Blue text marks prompt hints which are inserted to condition the model's response. These can be in-context examples, or the "hard-coded start" of the model's completion. For example, in the Q/A prompt template, we append `Answer:` as a hard-coded start for the model's completion to influence its response to answer the question marked with `Question:`.
- Green text marks the area where the model will generate a completion.

B. Basic prompt templates

Here we list the prompt templates we used which do not incorporate any examples for in-context learning. These queries can be used in n-shot prompts to boost the performance or induce a certain behavior from the model.

LISTING 1. Basic prompt template

```
Is the following function buggy? Please answer Yes or No.
```
[Code]
```
[Completion]
```

LISTING 2. Basic template with system prompt

```
<SystemToken>I want you to act as a vulnerability detection system.</SystemToken>
Is the following function buggy? Please answer Yes or No.
```
[Code]
```
[Completion]
```

LISTING 3. CWE list query (all bug types in the dataset)

```
Does the following function contain one of the following bug types? Please answer Yes or No.
- CWE-190: Integer Overflow
- CWE-476: Null Pointer Dereference
- CWE-125: Out-of-bound Read
- CWE-787: Out-of-bound Write
- CWE-416: Use After Free
```
[Code]
```
[Completion]
```

LISTING 4. Q/A template

```
Question: Is the following function buggy? Please answer Yes or No.
```
[Code]
```
Answer: [Completion]
```

C. n-shot prompt templates

There are several dimensions to our design, which can affect the model's performance:

- **From which dataset to select the examples?** We selected examples from the SVEN evaluation dataset (excluding the example in the query and its correponding buggy or fixed version), or from the BigVul or D2A datasets (as baselines for our CoT approaches which depend on these datasets).
- **How to order the examples?** We have found that the model performs best when the order of the examples are randomly shuffled in the prompt template. We also experimented with placing vulnerable examples first, non-vulnerable examples first, or sorting by similarity score; in these strategies, the models may suffer from recency bias, as noted by Zhao et al. [73].

- **What is the distribution of labels in the examples?** We can adjust the distribution of labels to include more vulnerable or non-vulnerable examples. We found that a balanced distribution worked best in our experiments, possibly due to the majority label bias discussed in Zhao et al. [73].
- **How many examples are included in the context?** We experimented to find the best number of examples; the models generally performed better with more examples.
- **How are the examples selected?** In the basic version, we select examples randomly from the dataset.

The in-context examples can be selected randomly, or by looking up the most similar code in a labeled dataset according to a similarity score.

We used this template to induce chain-of-thought responses by replacing the example responses in blue with chains of thought (shown below).

The hyphens `---` denote breaks between the example responses and subsequent queries. For chat models, we sent each query and response to the model as a separate message in the appropriate role.

LISTING 5. n-shot prompt template

````
Is the following function buggy? Please answer Yes or No.
```
[Labeled Code 1]
```
Yes, the program is buggy.
---
Is the following function buggy? Please answer Yes or No.
```
[Labeled Code 2]
```
No, the program is not buggy.
---
Is the following function buggy? Please answer Yes or No.
```
[Unseen Code]
```
[Completion]
````

D. Chain-of-thought from CWE Descriptions

Example for in-context learning, sourced from https://nvd.nist.gov/vuln/detail/CVE-2013-6051:

LISTING 6. CoT-CVE in-context example

````
Is the following function buggy? Please answer Yes or No.
```
[Code]
```
The bgp_attr_unknown function in bgp_attr.c in Quagga 0.99.21 does not properly initialize the
    total variable, which allows remote attackers to cause a denial of service (bgpd crash) via
    a crafted BGP update. Therefore, Yes, the program is buggy.
````

E. Chain-of-thought from Static Analysis

Example for in-context learning, sourced from hypothetical output from the Infer static analyzer [10, 75]:

LISTING 7. CoT-SA in-context example

````
Is the following function buggy? Please answer Yes or No.
```
[Code]
```
1. A buffer buf of size 10 is allocated at line 1.
2. An index i is initialized to a value in the range [0, 100] at line 2.
3. The index i is used to access buf at line 3. This may exceed the bounds of buf.Therefore,
    Yes, the program is buggy.
````