update at end of Iteration 2 (2 weeks per iteration) per board:
https://github.com/orgs/CyberSecAI/projects/2/views/1?pane=issue&itemId=76369142

Summary

1. The main components have been built and will be tested in future iterations with a proposal to apply to the subset of CISA ADP Vulnrichment CWE assignments.
2. The User Scenarios need to be defined as this will determine everything from Architecture, APIs, Deployment.

Details

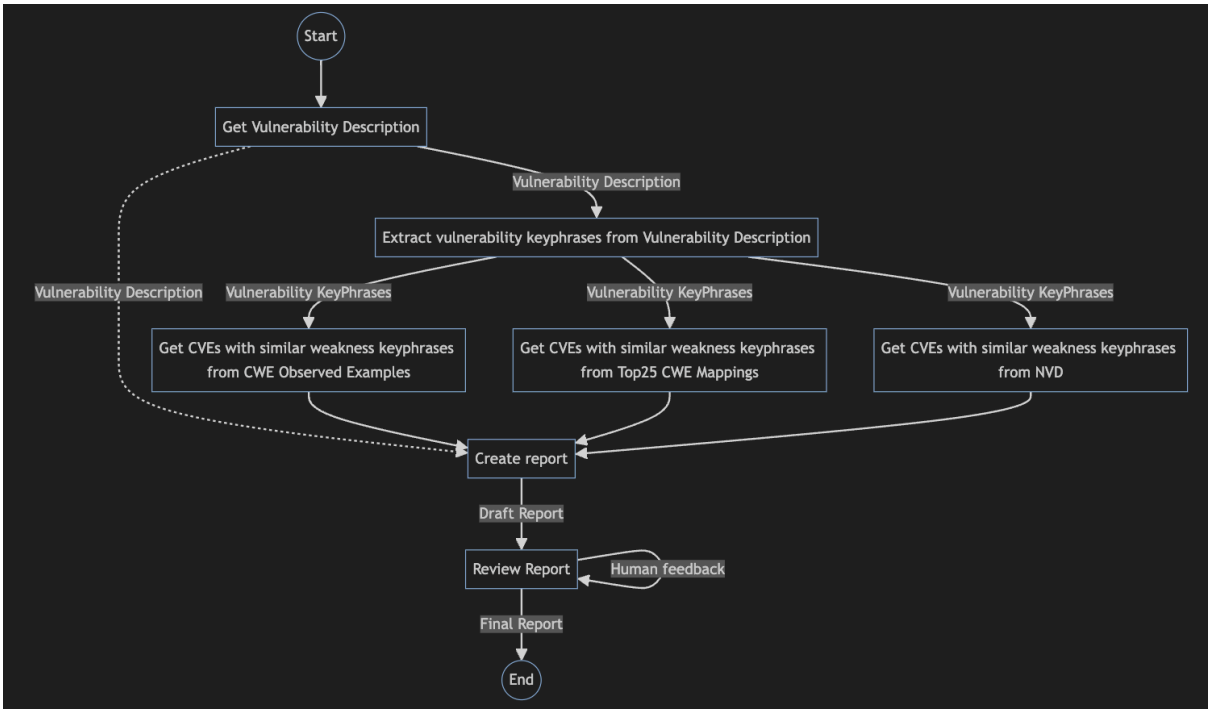
No progress on Capture User Scenarios

https://github.com/orgs/CyberSecAI/projects/2/views/1?pane=issue&itemId=76369142. We discussed meeting to cover this - but didn't.

1. Independent of this, as a first run in the wild of the current solution, I propose to apply the solution to CISA ADP Vulnrichment for assigned CWEs (where I applied an earlier less capable solution https://cybersecai.github.io/Vulnrichment/Vulnrichment/) to create Github Issues for identified mis-assigned CWEs. I'll let Lindsey Cerkovnik (CISA) know.
    1. The solution does not need to be hosted for this i.e. it can run locally - calling LLMs remotely via APIs.
    2. It will produce a Markdown report - that will include the parts shown below (and the CWE assignment and rationale as before)
2. This will dictate the solution(s).

Architecture Components Status

The main architecture components have been built (as a Multi-Agent System) and are being tested. See diagram below / here https://cybersecai.github.io/agents/Build_agents/#architecture :



NVD Reference links

Discussion on Not doing data extraction from NVD Reference links https://github.com/orgs/CyberSecAI/projects/2/views/1?pane=issue&itemId=74774941 with data input from Jerry Gamblin

CWE Vulnerability Report Example

See an example output for the new parts below

1. Vulnerability KeyPhrase Extraction
2. Similar CVEs from the NVD
    a. The similarity is primarily based on the weakness but also includes the Vendor/Product.
3. Similar CVEs from Top 25 CWE Mappings

The root cause CWE text is in **bold** in all places it occurs.
Github hyperlinks all CVE references automatically in the markdown to GHSA.

Vulnerability Report

Original Vulnerability Description

The Cisco Discovery Protocol implementation in Cisco IOS XR Software does not do **improper validation of string input** from certain fields which could allow an unauthenticated, adjacent attacker to execute arbitrary code or cause a reload on an affected device. The vulnerability is due to **improper validation of string input** from certain fields in Cisco Discovery Protocol messages. An attacker could exploit this vulnerability by sending a malicious Cisco Discovery Protocol packet to an affected device.

Extracted Key Phrases

Category	Description
[WEAKNESS]	**Improper validation of string input**
[PRODUCT]	Cisco Discovery Protocol implementation in Cisco IOS XR Software
[VERSION]	Not specified
[ATTACKER]	Unauthenticated, adjacent attacker
[IMPACT]	Execute arbitrary code or cause a reload on an affected device
[VECTOR]	Sending a malicious Cisco Discovery Protocol packet to an affected device
[ROOTCAUSE]	**Improper validation of string input** from certain fields in Cisco Discovery Protocol messages

Similar CVEs from the NVD

CWE-ID	CWE-Description	CVE-ID	CVE-Description
CWE-78	OS Command Injection	CVE-2017-12243	A vulnerability in the Cisco Unified Computing System (UCS) Manager, Cisco Firepower 4100 Series Next-Generation Firewall (NGFW), and Cisco Firepower 9300 Security Appliance could allow an authenticated, local attacker to obtain root shell privileges on the device, aka Command Injection. The vulnerability is due to **improper validation of string input** in the shell application. An attacker could exploit this vulnerability through the use of malicious commands. A successful exploit could allow the attacker to obtain root shell privileges on the device.
CWE-20	Improper Input Validation	CVE-2019-1831	A vulnerability in the email message scanning of Cisco AsyncOS Software for Cisco Email Security Appliance (ESA) could allow an unauthenticated, remote attacker to bypass configured content filters on the device. The vulnerability is due to **improper input validation** of the email body. An attacker could exploit this vulnerability by inserting specific character strings in the message. A successful exploit could allow the attacker to bypass configured content filters that would normally drop the email.
CWE-787, CWE-134	Out-of-bounds Write, Use of Externally-Controlled Format String	CVE-2020-3118	A vulnerability in the Cisco Discovery Protocol implementation for Cisco IOS XR Software could allow an unauthenticated, adjacent attacker to execute arbitrary code or cause a reload on an affected device. The vulnerability is due to **improper validation of string input** from certain fields in Cisco Discovery Protocol messages. An attacker could exploit this vulnerability by sending a malicious Cisco Discovery Protocol packet to an affected device. A successful exploit could allow the attacker to cause a stack overflow, which could allow the attacker to execute arbitrary code with administrative privileges on an affected device.

Similar CVEs from Top 25 CWE Mappings

CWE-ID	CWE-Description	CVE-ID	CVE-Description
CWE-20	Improper Input Validation	CVE-2021-0322	In onCreate of SlicePermissionActivity.java, there is a possible misleading string displayed due to **improper input validation**. This could lead to local information disclosure with User execution privileges needed. User interaction is needed for exploitation.
CWE-704	Incorrect Type Conversion or Cast	CVE-2021-28918	**Improper input validation** of octal strings in netmask npm package v1.0.6 and below allows unauthenticated remote attackers to perform indeterminate SSRF, RFI, and LFI attacks on many of the dependent packages. A remote unauthenticated attacker can bypass packages relying on netmask to filter IPs and reach critical VPN or LAN hosts.

The CWE assignment and rationale would also be added - this part exists already and is not shown. See https://github.com/CyberSecAI/CWEMap/blob/main/demos/CWE_Model_Demos_Iteration_1.pdf. But it would now use the additional data sources above to inform the assignment.

Details on Key Phrase Extraction

Based on the conversation in the last meeting on https://github.com/CyberSecAI/CWEMap/issues/6#issue-2466191165, and in decomposing the problem in general into smaller parts,we needed a second component that extracts the keyphrases from a vulnerability description.

see example output in https://github.com/CyberSecAI/CWEMap/blob/main/prompts/extract_key_entities/README.md

- The [ROOTCAUSE] can be extracted for any description - it's only shown in the first example.
- I also need to specify that there can be more than one instance of some of the labels

I have previously built other entity extraction (vulnerability keyphrase extraction) solutions using non-LLM technology (regex, fuzzy matching, Spacy NER, BERT-based models like GliNER... etc....).

- I gave a talk on this https://youtu.be/AtN_tav_5Mc?list=PLhgFPf60wAApHzd6mTmwN7lQ8V_ZiCZfu&t=1165
- MITRE CWE use a regex tool AFAIK

But I think an LLM solution should be used here because there's a lot of known-good data and guidance that an LLM can best take advantage of (that other methods can't) and also some nuance

1. https://www.cve.org/Resources/General/Key-Details-Phrasing.pdf
2. https://cwe.mitre.org/documents/cwe_usage/guidance.html
3. The Top25 dataset