From Idea to Open-Source: Building CNA-GURU

A Generative AI Assistant for Security Advisories

Ryan Nolette

AWS Security

$whoami

Ryan Nolette

Senior Security Engineer, AWS Security Outreach
President, HTCIA New England Chapter

@sonofaglitch

Coauthor of **Huntpedia**

Coauthor of Amazon Detective

Get in contact

Agenda

- $whoami
- Inspiration from VulnCon 2024
- Conceptualizing CNA-GURU
- Building CNA-GURU
- Key Features and Functionality
- Evaluation and Accuracy
- Benefits and Use Cases
- Open-Sourcing and Community Engagement
- Conclusion

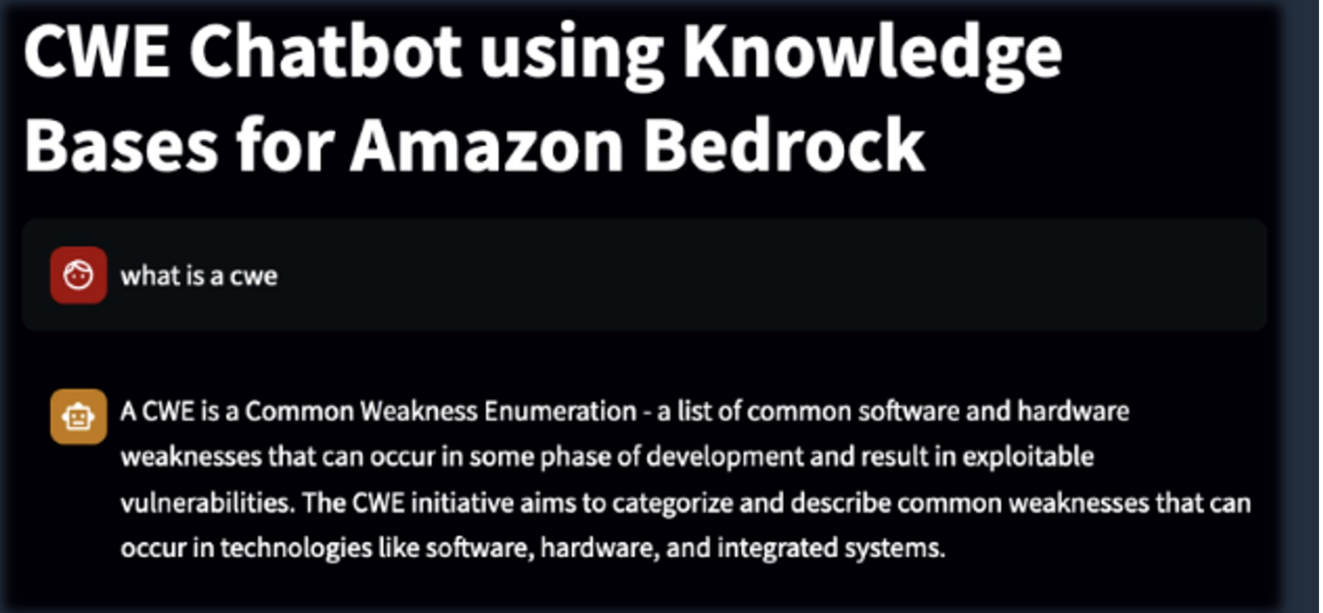
Inspiration from VulnCon 2024

- Official name "Chatbot-to-help-security-teams-perform-vulnerability-assessments "
- Nicknames "CNA-GURU" and "ChatCVE"
- Discussions with security professionals about the challenges of managing security advisories
- Increase in CVE creation from 2024 rules update
- Identifying the common pain points and the need for a solution

aws security

Conceptualizing CNA-GURU

- Super scalable
- Statistically significant accuracy in responses.
- No model training required
- Extendable and flexible
- Not overly expensive
- Private



CWE Chatbot using Knowledge Bases for Amazon Bedrock

what is a cwe

A CWE is a Common Weakness Enumeration - a list of common software and hardware weaknesses that can occur in some phase of development and result in exploitable vulnerabilities. The CWE initiative aims to categorize and describe common weaknesses that can occur in technologies like software, hardware, and integrated systems.

aws security

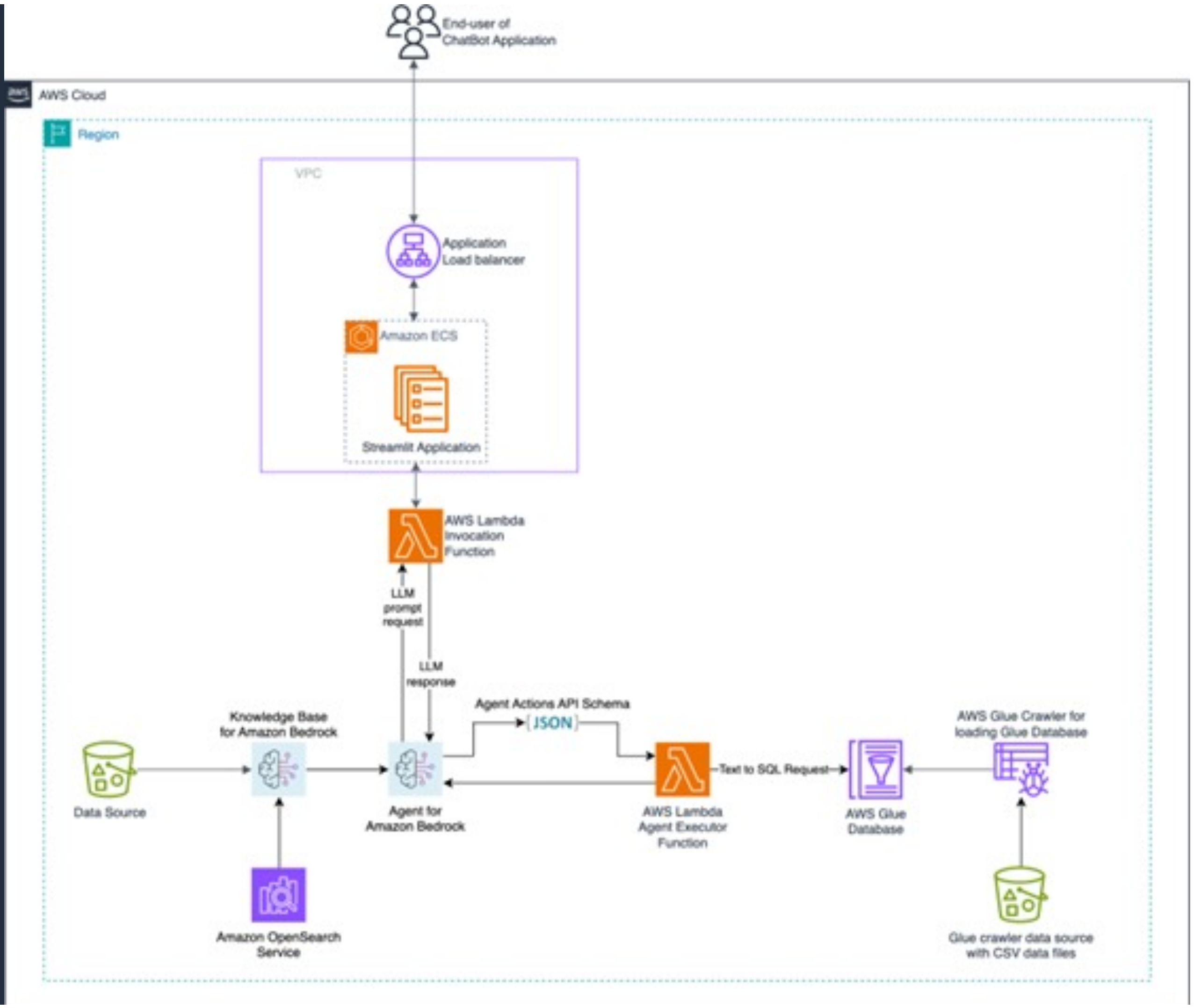
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Building CNA-GURU

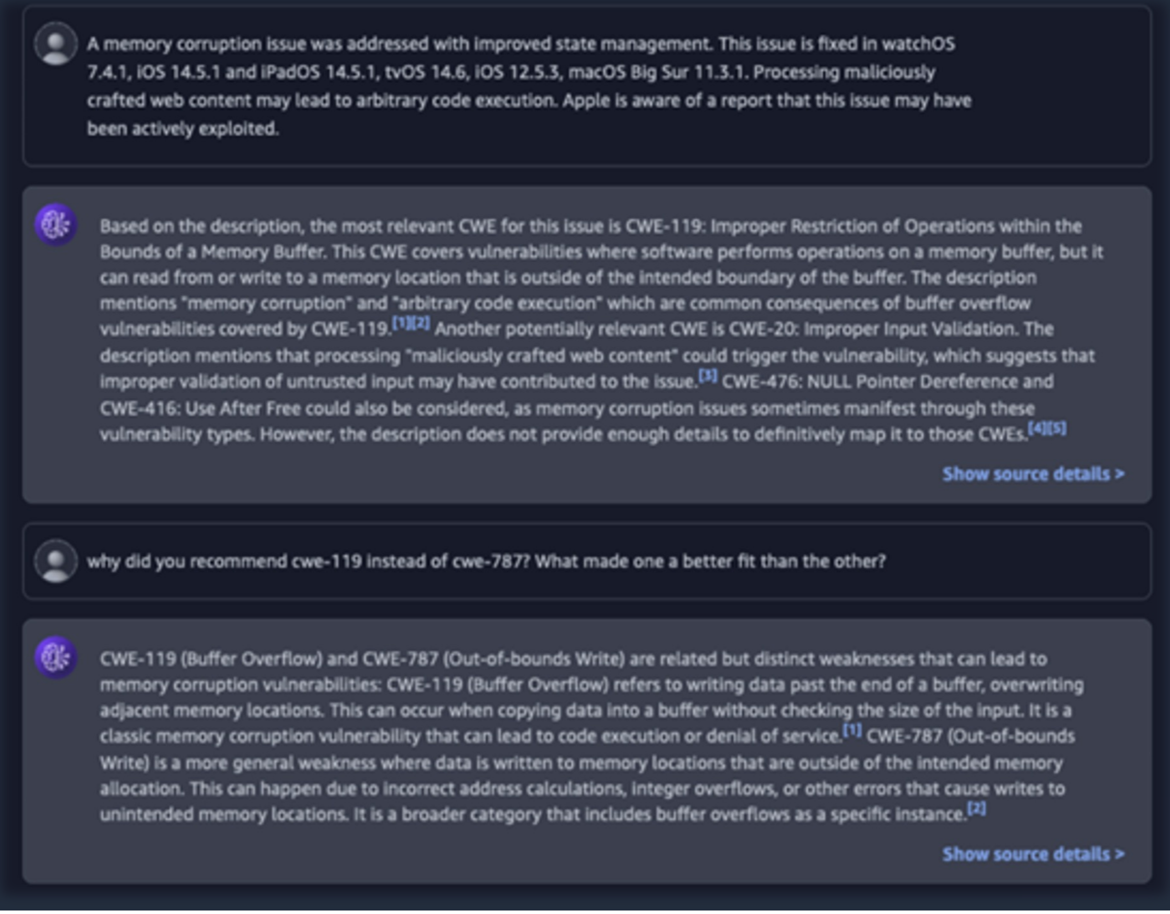
Target technology stack:

- Amazon Bedrock
- Amazon OpenSearch Serverless
- Amazon ECS
- AWS Glue
- AWS Lambda
- Amazon S3
- Amazon Athena
- Elastic Load Balancer

aws security


End-user of ChatBot Application
AWS Cloud
Region
VPC
Application Load balancer
Amazon ECS
Streamlit Application
AWS Lambda Invocation Function
LLM prompt request
LLM response
Agent Actions API Schema
JSON
Knowledge Base for Amazon Bedrock
Data Source
Agent for Amazon Bedrock
AWS Lambda Agent Executor Function
Text to SQL Request
AWS Glue Database
AWS Glue Crawler for loading Glue Database
Amazon OpenSearch Service
Glue crawler data source with CSV data files

Key Features and Functionality: CWE support



A memory corruption issue was addressed with improved state management. This issue is fixed in watchOS 7.4.1, iOS 14.5.1 and iPadOS 14.5.1, tvOS 14.6, iOS 12.5.3, macOS Big Sur 11.3.1. Processing maliciously crafted web content may lead to arbitrary code execution. Apple is aware of a report that this issue may have been actively exploited.

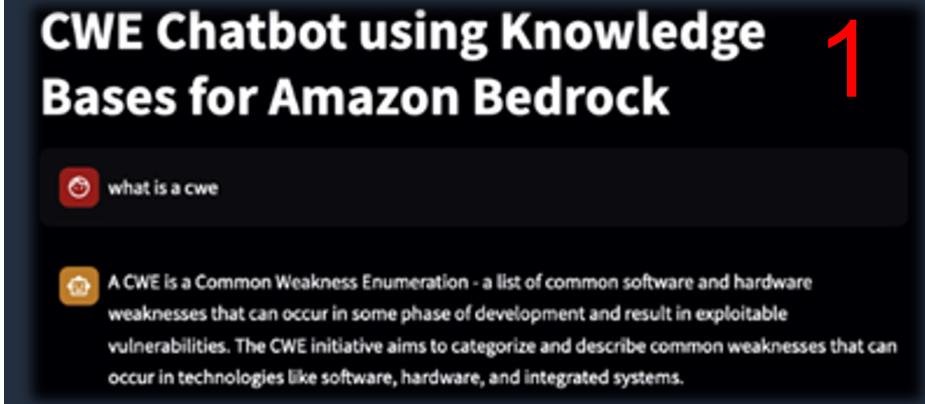
Based on the description, the most relevant CWE for this issue is CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer. This CWE covers vulnerabilities where software performs operations on a memory buffer, but it can read from or write to a memory location that is outside of the intended boundary of the buffer. The description mentions "memory corruption" and "arbitrary code execution" which are common consequences of buffer overflow vulnerabilities covered by CWE-119.[1][2] Another potentially relevant CWE is CWE-20: Improper Input Validation. The description mentions that processing "maliciously crafted web content" could trigger the vulnerability, which suggests that improper validation of untrusted input may have contributed to the issue.[3] CWE-476: NULL Pointer Dereference and CWE-416: Use After Free could also be considered, as memory corruption issues sometimes manifest through these vulnerability types. However, the description does not provide enough details to definitively map it to those CWEs.[4][5]

why did you recommend cwe-119 instead of cwe-787? What made one a better fit than the other?

CWE-119 (Buffer Overflow) and CWE-787 (Out-of-bounds Write) are related but distinct weaknesses that can lead to memory corruption vulnerabilities: CWE-119 (Buffer Overflow) refers to writing data past the end of a buffer, overwriting adjacent memory locations. This can occur when copying data into a buffer without checking the size of the input. It is a classic memory corruption vulnerability that can lead to code execution or denial of service.[1] CWE-787 (Out-of-bounds Write) is a more general weakness where data is written to memory locations that are outside of the intended memory allocation. This can happen due to incorrect address calculations, integer overflows, or other errors that cause writes to unintended memory locations. It is a broader category that includes buffer overflows as a specific instance.[2]

aws
security

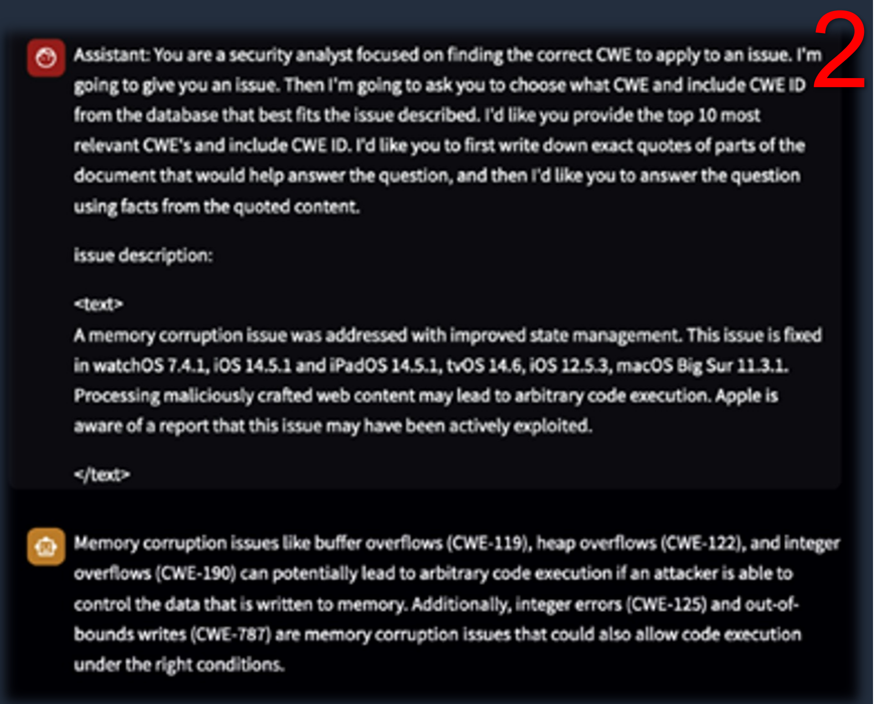
Key Features and Functionality



CWE Chatbot using Knowledge Bases for Amazon Bedrock

what is a cwe

A CWE is a Common Weakness Enumeration - a list of common software and hardware weaknesses that can occur in some phase of development and result in exploitable vulnerabilities. The CWE initiative aims to categorize and describe common weaknesses that can occur in technologies like software, hardware, and integrated systems.



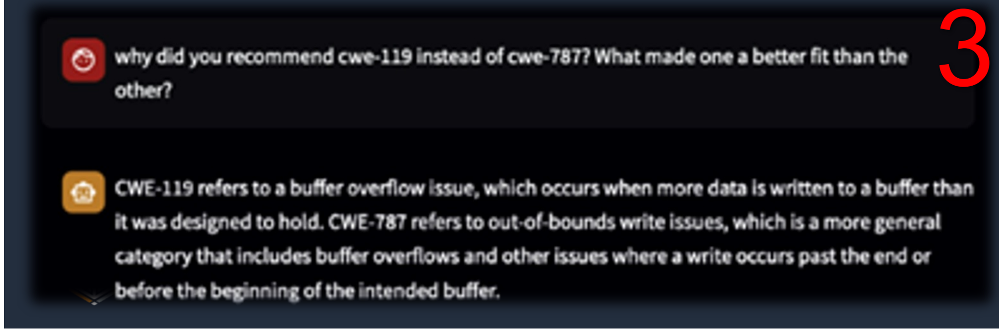
Assistant: You are a security analyst focused on finding the correct CWE to apply to an issue. I'm going to give you an issue. Then I'm going to ask you to choose what CWE and include CWE ID from the database that best fits the issue described. I'd like you provide the top 10 most relevant CWE's and include CWE ID. I'd like you to first write down exact quotes of parts of the document that would help answer the question, and then I'd like you to answer the question using facts from the quoted content.

issue description:

<text>
A memory corruption issue was addressed with improved state management. This issue is fixed in watchOS 7.4.1, iOS 14.5.1 and iPadOS 14.5.1, tvOS 14.6, iOS 12.5.3, macOS Big Sur 11.3.1. Processing maliciously crafted web content may lead to arbitrary code execution. Apple is aware of a report that this issue may have been actively exploited.

</text>

Memory corruption issues like buffer overflows (CWE-119), heap overflows (CWE-122), and integer overflows (CWE-190) can potentially lead to arbitrary code execution if an attacker is able to control the data that is written to memory. Additionally, integer errors (CWE-125) and out-of-bounds writes (CWE-787) are memory corruption issues that could also allow code execution under the right conditions.



why did you recommend cwe-119 instead of cwe-787? What made one a better fit than the other?

CWE-119 refers to a buffer overflow issue, which occurs when more data is written to a buffer than it was designed to hold. CWE-787 refers to out-of-bounds write issues, which is a more general category that includes buffer overflows and other issues where a write occurs past the end or before the beginning of the intended buffer.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Evaluation and Accuracy: Development Timeline & ROI Impact

Key Points:

- Total development time: 5hrs from concept to working MVP
- 2hrs: Initial AWS AI service research
- 3hrs: Python script development, knowledge base creation, and testing
- Zero additional AWS services required beyond Bedrock
- 90% accuracy across 300 tests of randomly selected CVEs

Business Impact:

- Significant time savings in CWE analysis
- Previous process: 3hrs per analysis
- Each analysis needed specialized knowledge in CVE, CWE, and vulnerability scoring

ROI Highlights:

- Time reduction: From 3hrs to seconds per analysis
- Training efficiency: Reduces dependency on extensive analyst training
- Immediate value: Positive ROI achieved after first analysis

aws security

Evaluation and Accuracy: Evolution of CNA-GURU

CNA-GURU (Original)

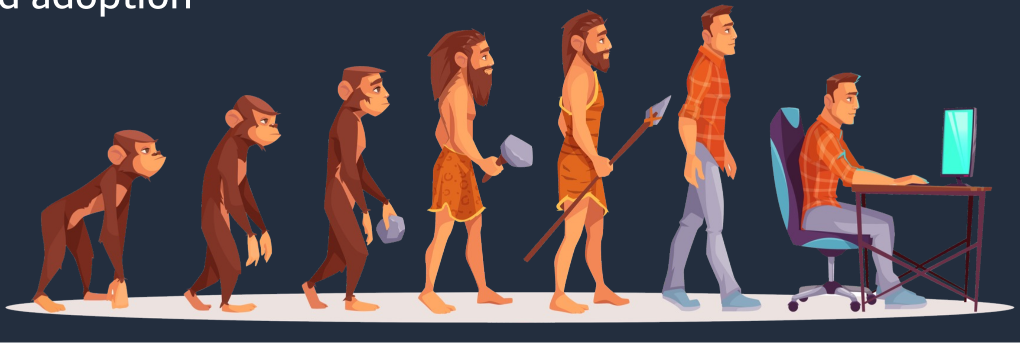
- Comprehensive but resource-intensive implementation
- Built with full AWS service stack
- Cost: Several hundred dollars/month
- High accuracy but expensive for widespread adoption

CNA-GURU JR

- Reduced infrastructure
- Leverages custom prompts rather than knowledge base
- Similar user experience to original version
- Cost: ~$20/month for team of 10

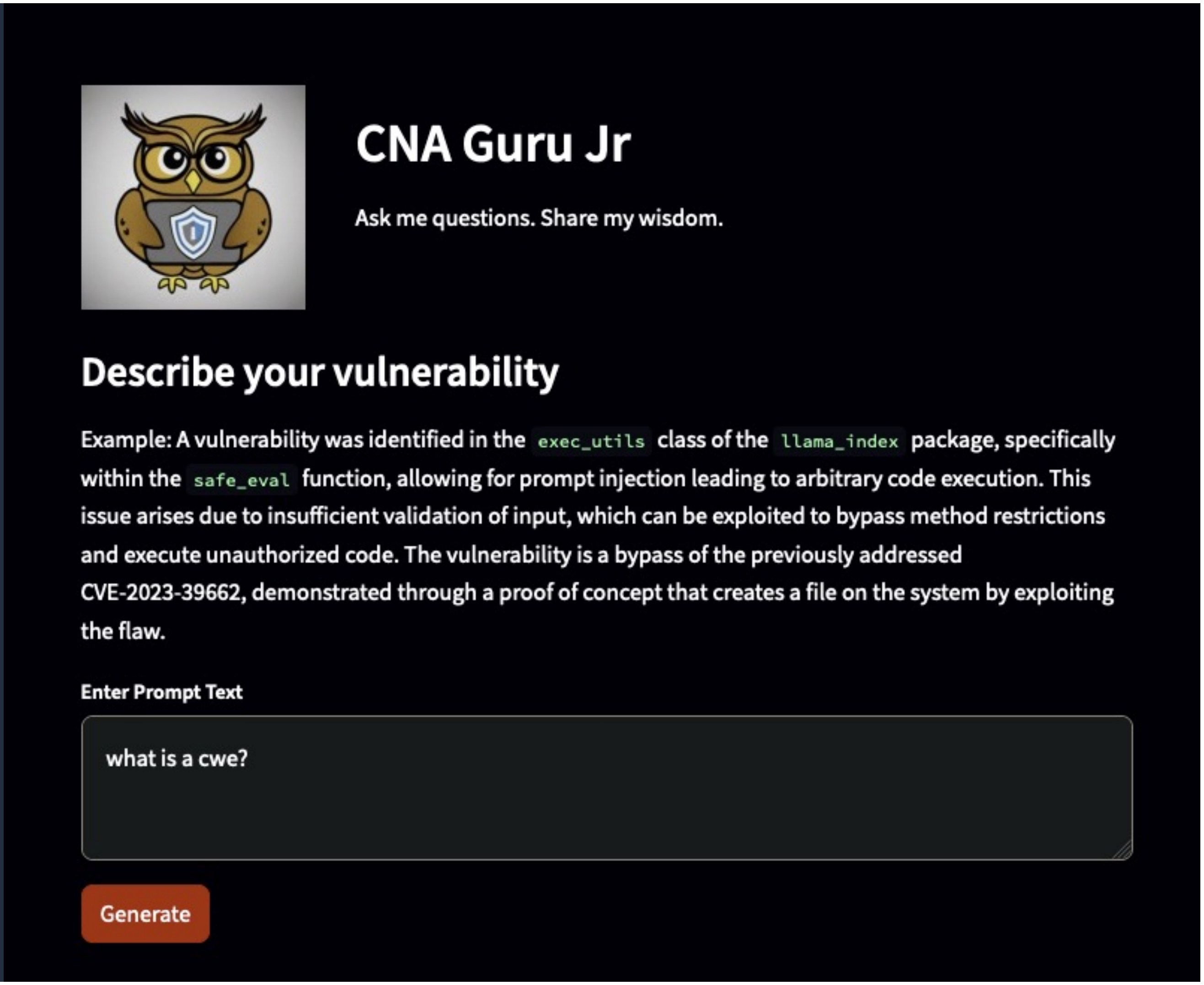
CNA-GURU Esq

- Runs in Jupyter notebook environment
- Interactive chatbot functionality
- No internet-facing infrastructure needed
- Cost: ~$20/month in production



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Evolution 1: reduce cost



CNA Guru Jr

Ask me questions. Share my wisdom.

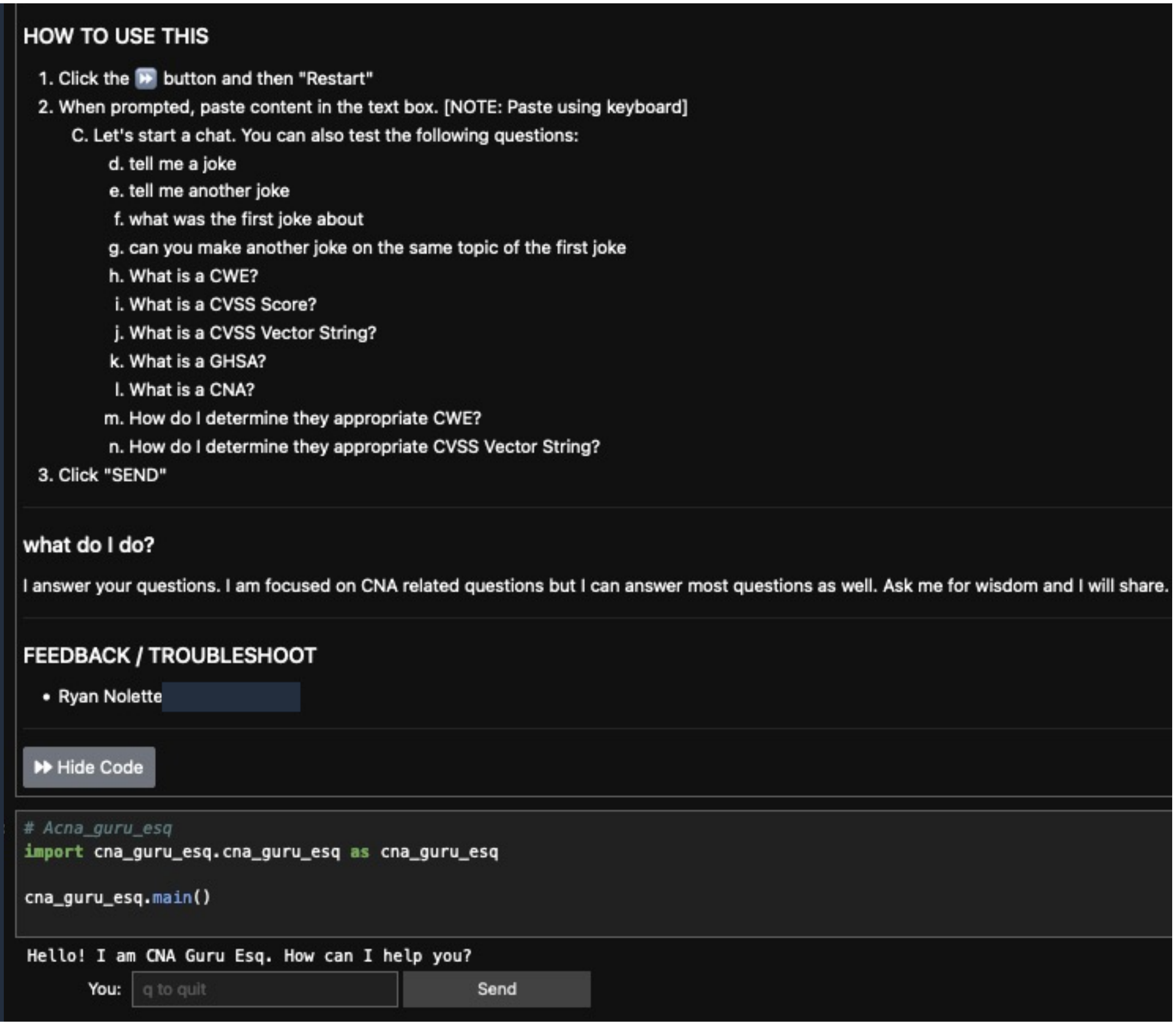
Describe your vulnerability

Example: A vulnerability was identified in the `exec_utils` class of the `llama_index` package, specifically within the `safe_eval` function, allowing for prompt injection leading to arbitrary code execution. This issue arises due to insufficient validation of input, which can be exploited to bypass method restrictions and execute unauthorized code. The vulnerability is a bypass of the previously addressed CVE-2023-39662, demonstrated through a proof of concept that creates a file on the system by exploiting the flaw.

Enter Prompt Text

what is a cwe?

Evolution 2: Jupyter notebook



HOW TO USE THIS

1. Click the ⏩ button and then "Restart"
2. When prompted, paste content in the text box. [NOTE: Paste using keyboard]
    C. Let's start a chat. You can also test the following questions:
        d. tell me a joke
        e. tell me another joke
        f. what was the first joke about
        g. can you make another joke on the same topic of the first joke
        h. What is a CWE?
        i. What is a CVSS Score?
        j. What is a CVSS Vector String?
        k. What is a GHSA?
        l. What is a CNA?
        m. How do I determine they appropriate CWE?
        n. How do I determine they appropriate CVSS Vector String?
3. Click "SEND"

what do I do?

I answer your questions. I am focused on CNA related questions but I can answer most questions as well. Ask me for wisdom and I will share.

FEEDBACK / TROUBLESHOOT

- Ryan Nolette [redacted]

```
# Acna_guru_esq
import cna_guru_esq.cna_guru_esq as cna_guru_esq

cna_guru_esq.main()
```

```
Hello! I am CNA Guru Esq. How can I help you?
```

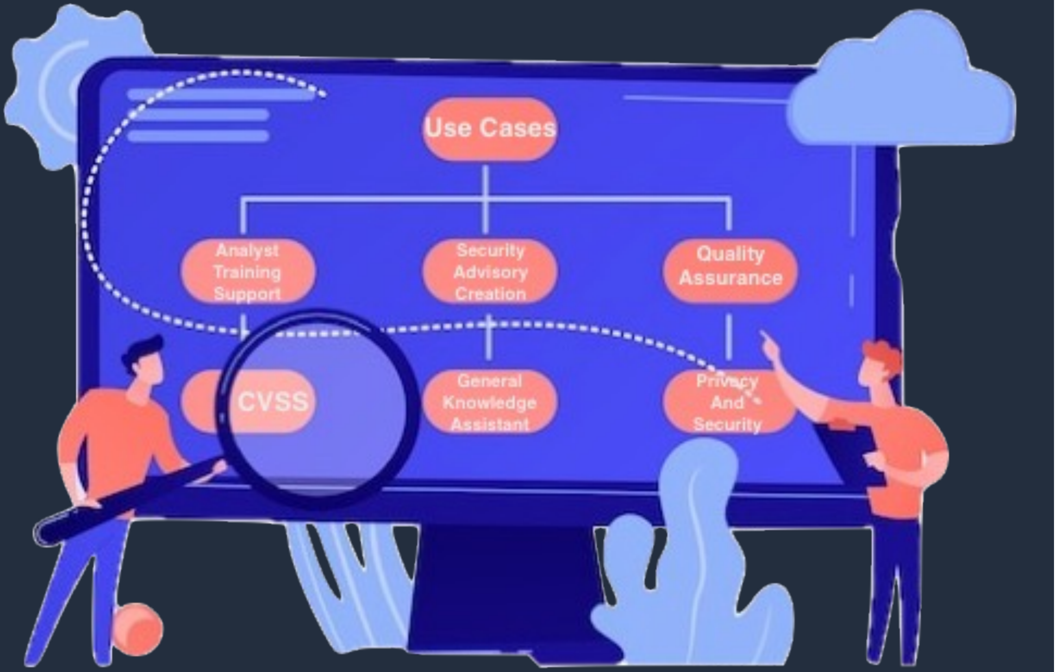
Benefits and Use Cases

Benefits

- Scalability
- Accessibility
- Privacy & Security
- Flexibility & Extensibility
- Accuracy & Efficiency
- Cost Effectiveness

Common Use Cases

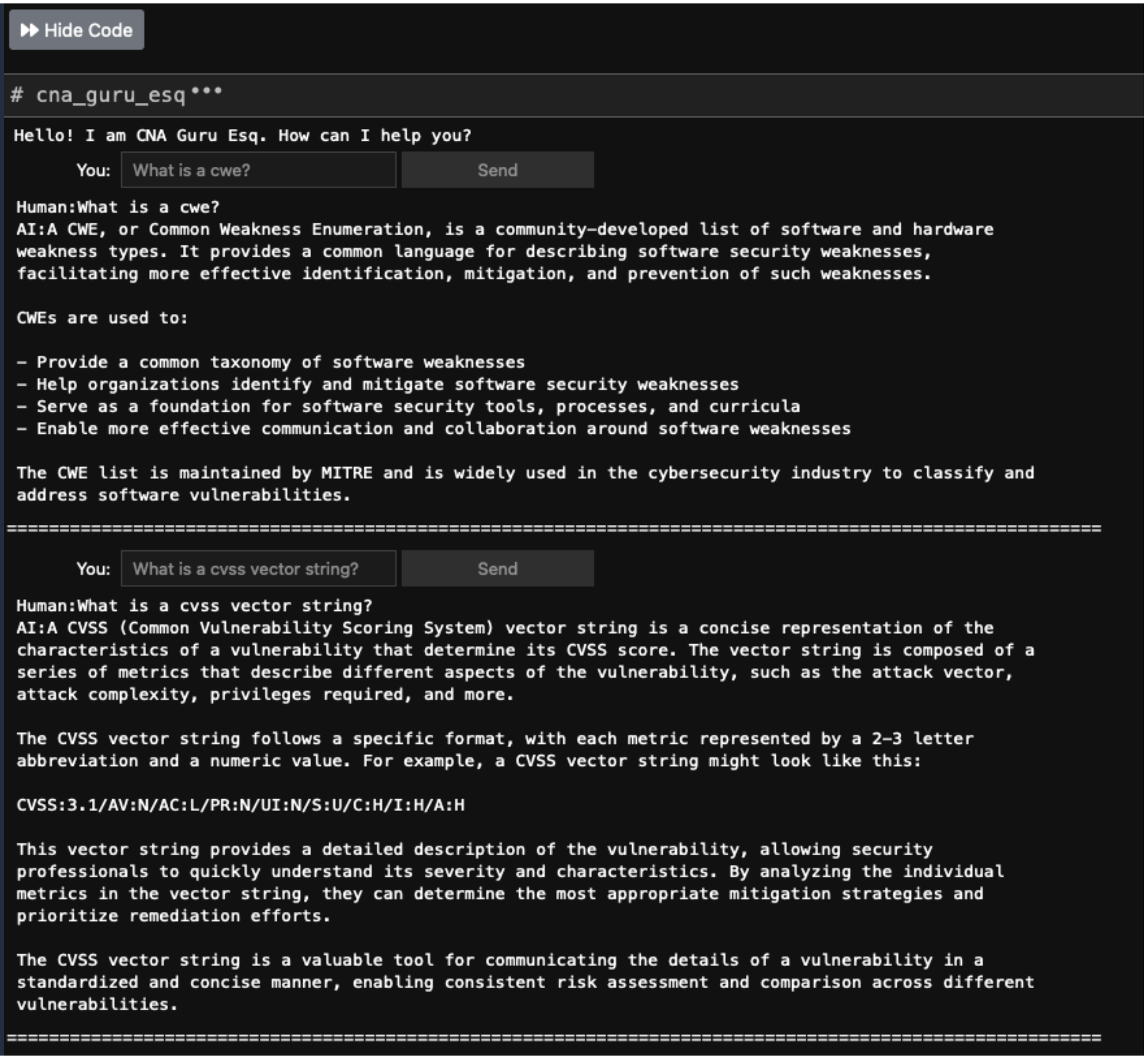
- Security Advisory Creation
- Analyst Training Support
- Quality Assurance



aws security

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Use Case:
CNA related
General knowledge



```
# cna_guru_esq
Hello! I am CNA Guru Esq. How can I help you?
```

You: What is a cwe?

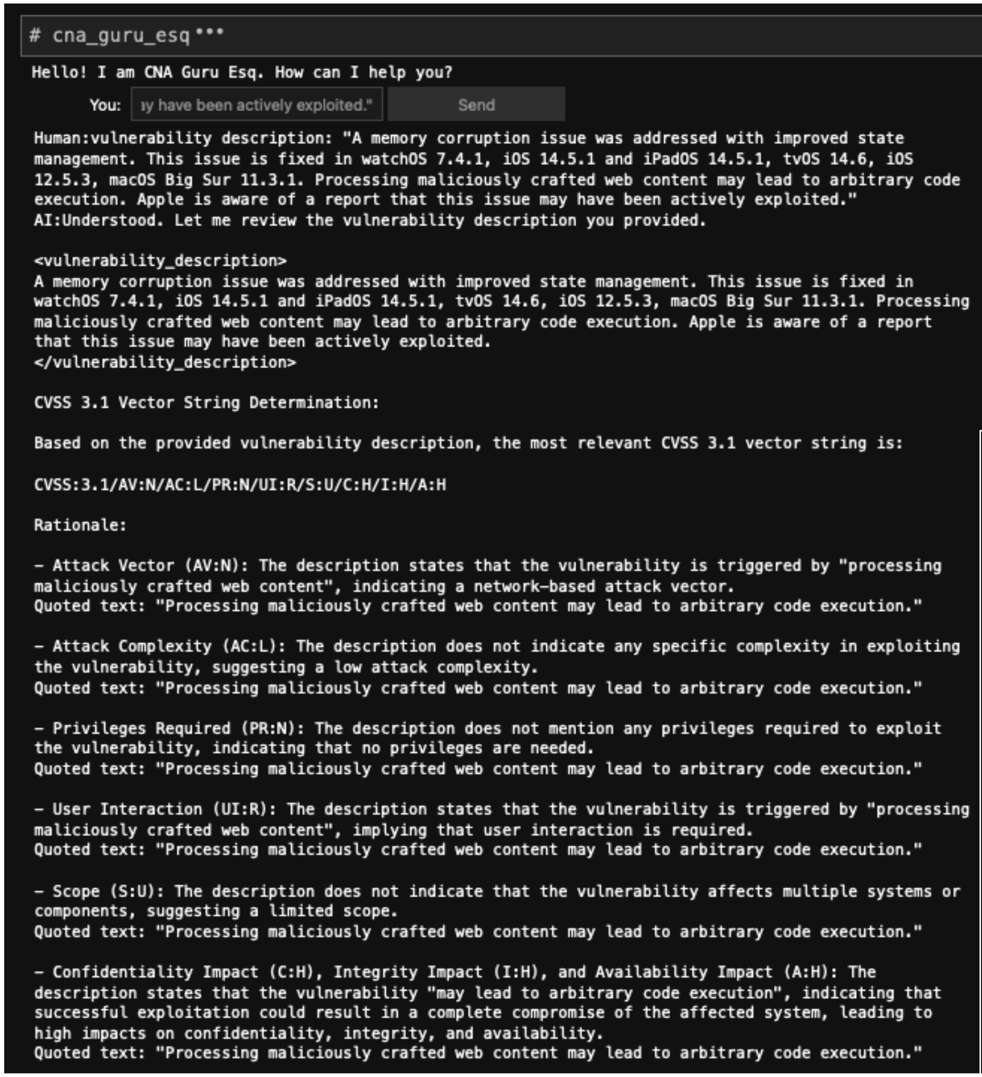
```
Human:What is a cwe?
AI:A CWE, or Common Weakness Enumeration, is a community-developed list of software and hardware
weakness types. It provides a common language for describing software security weaknesses,
facilitating more effective identification, mitigation, and prevention of such weaknesses.

CWEs are used to:

- Provide a common taxonomy of software weaknesses
- Help organizations identify and mitigate software security weaknesses
- Serve as a foundation for software security tools, processes, and curricula
- Enable more effective communication and collaboration around software weaknesses

The CWE list is maintained by MITRE and is widely used in the cybersecurity industry to classify and
address software vulnerabilities.
====================================================================================================
```

You: What is a cvss vector string?

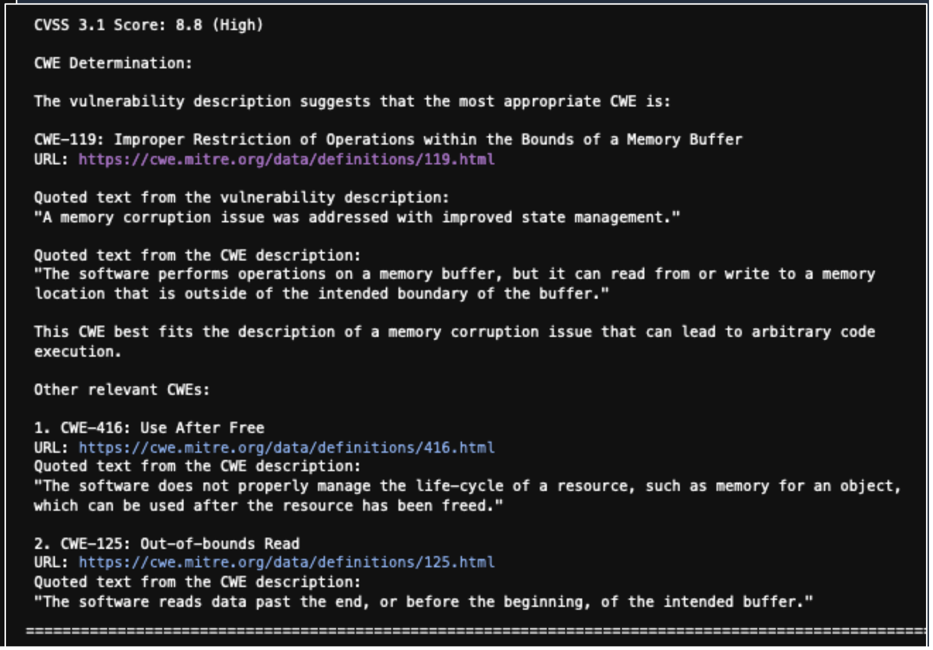
```
Human:What is a cvss vector string?
AI:A CVSS (Common Vulnerability Scoring System) vector string is a concise representation of the
characteristics of a vulnerability that determine its CVSS score. The vector string is composed of a
series of metrics that describe different aspects of the vulnerability, such as the attack vector,
attack complexity, privileges required, and more.

The CVSS vector string follows a specific format, with each metric represented by a 2-3 letter
abbreviation and a numeric value. For example, a CVSS vector string might look like this:

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

This vector string provides a detailed description of the vulnerability, allowing security
professionals to quickly understand its severity and characteristics. By analyzing the individual
metrics in the vector string, they can determine the most appropriate mitigation strategies and
prioritize remediation efforts.

The CVSS vector string is a valuable tool for communicating the details of a vulnerability in a
standardized and concise manner, enabling consistent risk assessment and comparison across different
vulnerabilities.
====================================================================================================
```

aws security

Use Case:
Calculate CVSS Score
Determine CWE

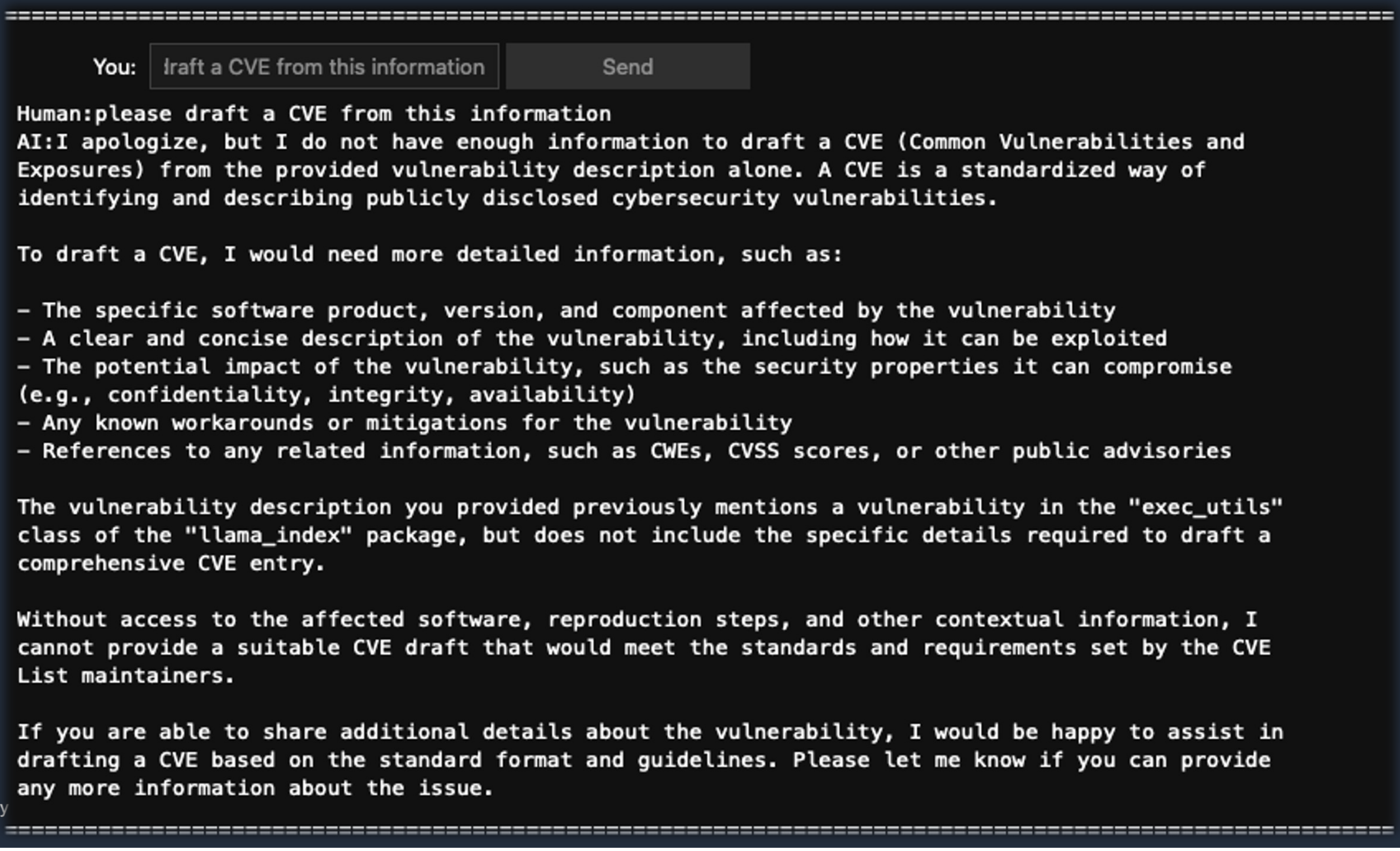


```
Hello! I am CNA Guru Esq. How can I help you?

Human:vulnerability description: "A memory corruption issue was addressed with improved state
management. This issue is fixed in watchOS 7.4.1, iOS 14.5.1 and iPadOS 14.5.1, tvOS 14.6, iOS
12.5.3, macOS Big Sur 11.3.1. Processing maliciously crafted web content may lead to arbitrary code
execution. Apple is aware of a report that this issue may have been actively exploited."
AI:Understood. Let me review the vulnerability description you provided.

<vulnerability_description>
A memory corruption issue was addressed with improved state management. This issue is fixed in
watchOS 7.4.1, iOS 14.5.1 and iPadOS 14.5.1, tvOS 14.6, iOS 12.5.3, macOS Big Sur 11.3.1. Processing
maliciously crafted web content may lead to arbitrary code execution. Apple is aware of a report
that this issue may have been actively exploited.
</vulnerability_description>

CVSS 3.1 Vector String Determination:

Based on the provided vulnerability description, the most relevant CVSS 3.1 vector string is:

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

Rationale:

- Attack Vector (AV:N): The description states that the vulnerability is triggered by "processing
maliciously crafted web content", indicating a network-based attack vector.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."

- Attack Complexity (AC:L): The description does not indicate any specific complexity in exploiting
the vulnerability, suggesting a low attack complexity.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."

- Privileges Required (PR:N): The description does not mention any privileges required to exploit
the vulnerability, indicating that no privileges are needed.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."

- User Interaction (UI:R): The description states that the vulnerability is triggered by "processing
maliciously crafted web content", implying that user interaction is required.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."

- Scope (S:U): The description does not indicate that the vulnerability affects multiple systems or
components, suggesting a limited scope.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."

- Confidentiality Impact (C:H), Integrity Impact (I:H), and Availability Impact (A:H): The
description states that the vulnerability "may lead to arbitrary code execution", indicating that
successful exploitation could result in a complete compromise of the affected system, leading to
high impacts on confidentiality, integrity, and availability.
Quoted text: "Processing maliciously crafted web content may lead to arbitrary code execution."
```



```
CVSS 3.1 Score: 8.8 (High)

CWE Determination:

The vulnerability description suggests that the most appropriate CWE is:

CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer
URL: https://cwe.mitre.org/data/definitions/119.html

Quoted text from the vulnerability description:
"A memory corruption issue was addressed with improved state management."

Quoted text from the CWE description:
"The software performs operations on a memory buffer, but it can read from or write to a memory
location that is outside of the intended boundary of the buffer."

This CWE best fits the description of a memory corruption issue that can lead to arbitrary code
execution.

Other relevant CWEs:

1. CWE-416: Use After Free
URL: https://cwe.mitre.org/data/definitions/416.html
Quoted text from the CWE description:
"The software does not properly manage the life-cycle of a resource, such as memory for an object,
which can be used after the resource has been freed."

2. CWE-125: Out-of-bounds Read
URL: https://cwe.mitre.org/data/definitions/125.html
Quoted text from the CWE description:
"The software reads data past the end, or before the beginning, of the intended buffer."

==============================================================================================
```

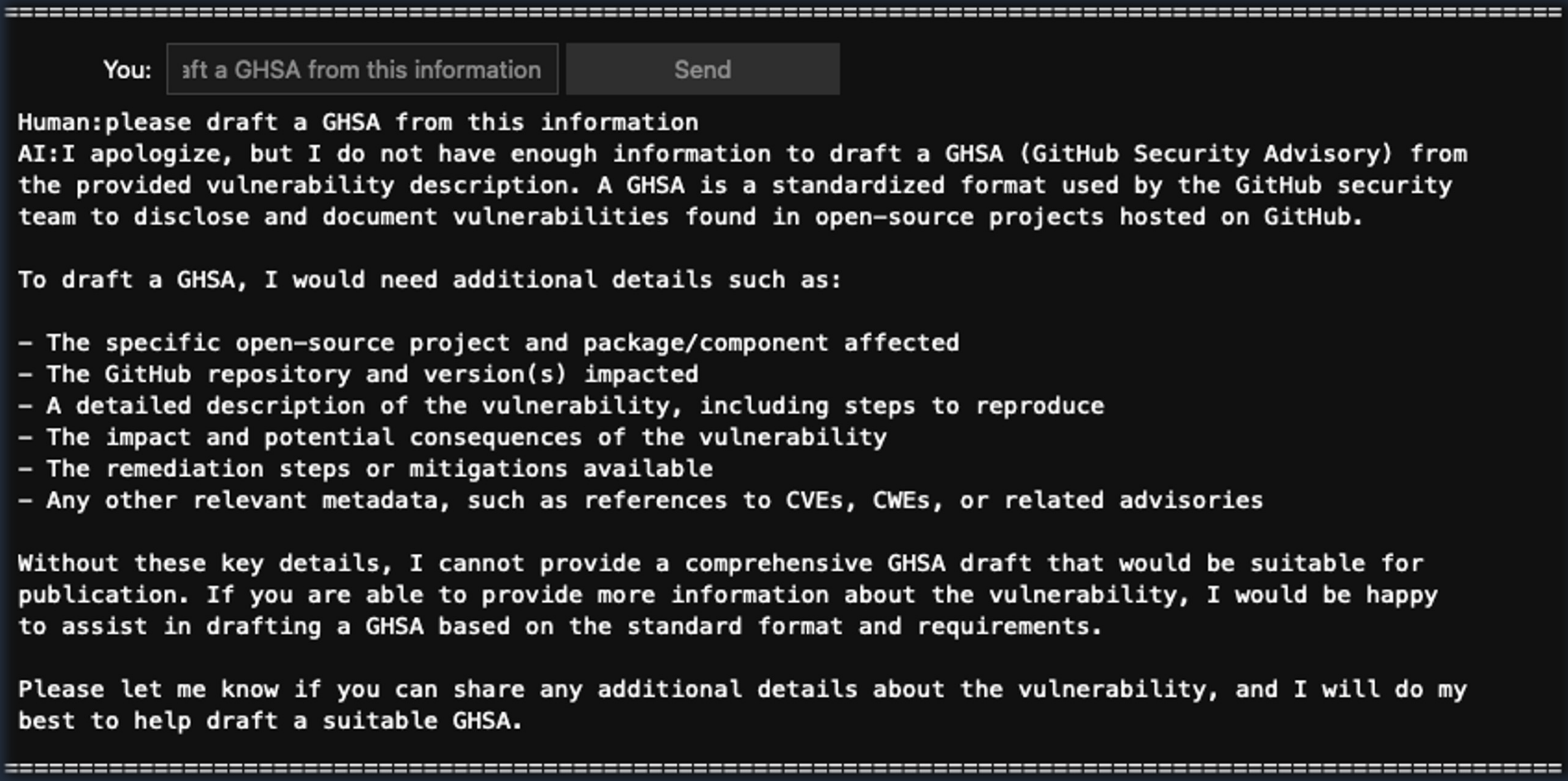
Use Case: Assistance with creating a CVE



lraft a CVE from this information

```
Human:please draft a CVE from this information
AI:I apologize, but I do not have enough information to draft a CVE (Common Vulnerabilities and
Exposures) from the provided vulnerability description alone. A CVE is a standardized way of
identifying and describing publicly disclosed cybersecurity vulnerabilities.

To draft a CVE, I would need more detailed information, such as:

- The specific software product, version, and component affected by the vulnerability
- A clear and concise description of the vulnerability, including how it can be exploited
- The potential impact of the vulnerability, such as the security properties it can compromise
(e.g., confidentiality, integrity, availability)
- Any known workarounds or mitigations for the vulnerability
- References to any related information, such as CWEs, CVSS scores, or other public advisories

The vulnerability description you provided previously mentions a vulnerability in the "exec_utils"
class of the "llama_index" package, but does not include the specific details required to draft a
comprehensive CVE entry.

Without access to the affected software, reproduction steps, and other contextual information, I
cannot provide a suitable CVE draft that would meet the standards and requirements set by the CVE
List maintainers.

If you are able to share additional details about the vulnerability, I would be happy to assist in
drafting a CVE based on the standard format and guidelines. Please let me know if you can provide
any more information about the issue.
```

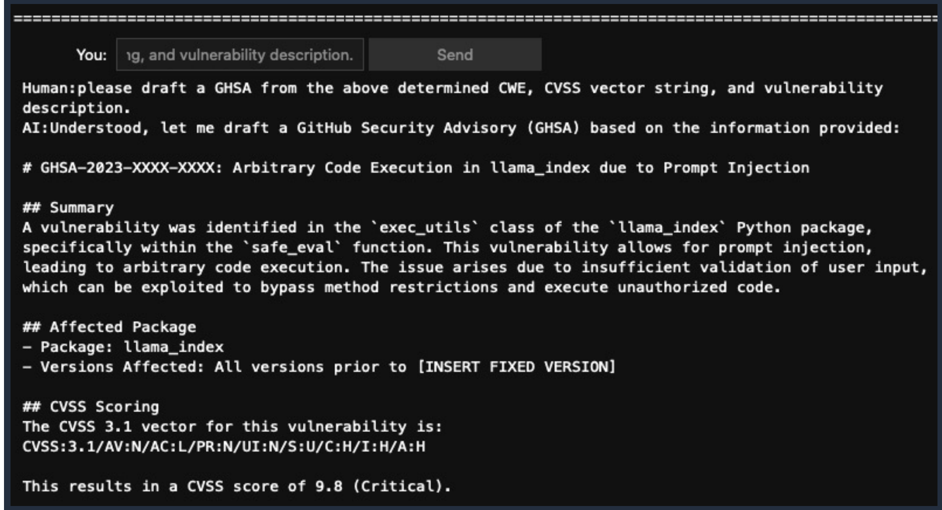
Use Case: Assistance with creating a GHSA



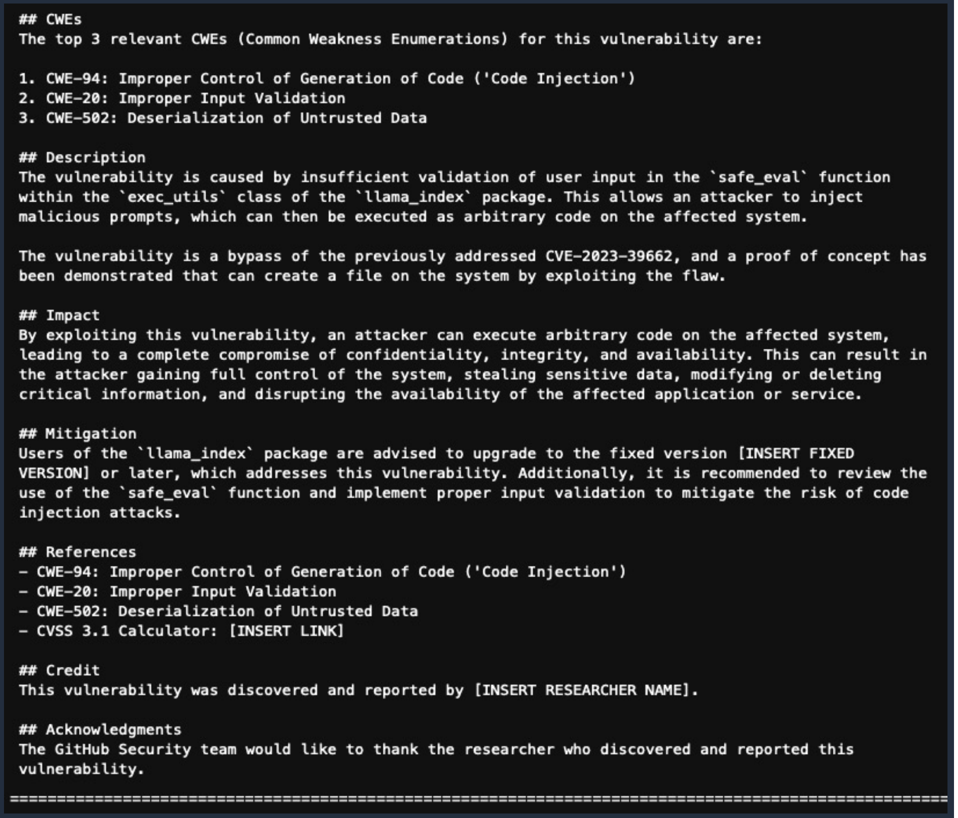
```
================================================================================================================
Human:please draft a GHSA from this information
AI:I apologize, but I do not have enough information to draft a GHSA (GitHub Security Advisory) from
the provided vulnerability description. A GHSA is a standardized format used by the GitHub security
team to disclose and document vulnerabilities found in open-source projects hosted on GitHub.

To draft a GHSA, I would need additional details such as:

- The specific open-source project and package/component affected
- The GitHub repository and version(s) impacted
- A detailed description of the vulnerability, including steps to reproduce
- The impact and potential consequences of the vulnerability
- The remediation steps or mitigations available
- Any other relevant metadata, such as references to CVEs, CWEs, or related advisories

Without these key details, I cannot provide a comprehensive GHSA draft that would be suitable for
publication. If you are able to provide more information about the vulnerability, I would be happy
to assist in drafting a GHSA based on the standard format and requirements.

Please let me know if you can share any additional details about the vulnerability, and I will do my
best to help draft a suitable GHSA.
================================================================================================================
```

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Use Case:
Create GHSA



```
==============================================================================================

Human:please draft a GHSA from the above determined CWE, CVSS vector string, and vulnerability
description.
AI:Understood, let me draft a GitHub Security Advisory (GHSA) based on the information provided:

# GHSA-2023-XXXX-XXXX: Arbitrary Code Execution in llama_index due to Prompt Injection

## Summary
A vulnerability was identified in the `exec_utils` class of the `llama_index` Python package,
specifically within the `safe_eval` function. This vulnerability allows for prompt injection,
leading to arbitrary code execution. The issue arises due to insufficient validation of user input,
which can be exploited to bypass method restrictions and execute unauthorized code.

## Affected Package
- Package: llama_index
- Versions Affected: All versions prior to [INSERT FIXED VERSION]

## CVSS Scoring
The CVSS 3.1 vector for this vulnerability is:
CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

This results in a CVSS score of 9.8 (Critical).
```

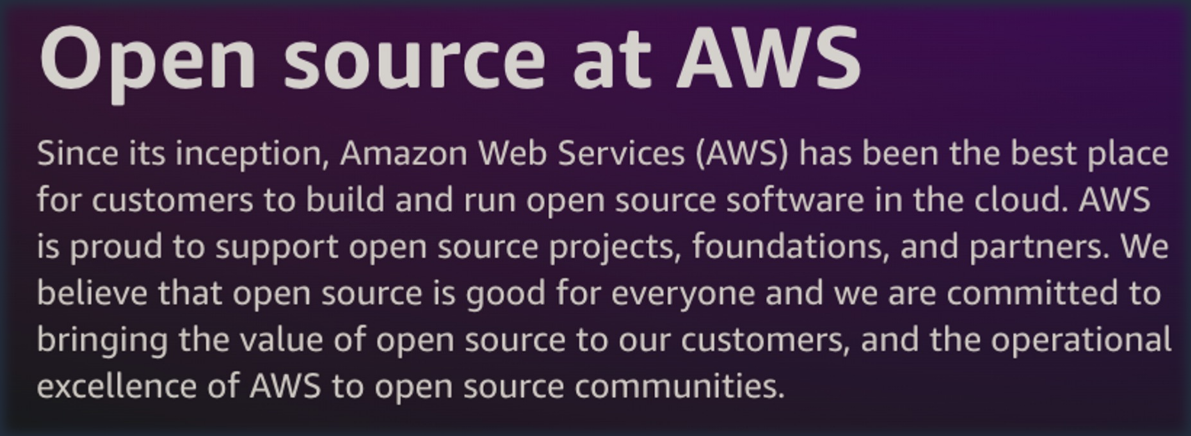


```
## CWEs
The top 3 relevant CWEs (Common Weakness Enumerations) for this vulnerability are:

1. CWE-94: Improper Control of Generation of Code ('Code Injection')
2. CWE-20: Improper Input Validation
3. CWE-502: Deserialization of Untrusted Data

## Description
The vulnerability is caused by insufficient validation of user input in the `safe_eval` function
within the `exec_utils` class of the `llama_index` package. This allows an attacker to inject
malicious prompts, which can then be executed as arbitrary code on the affected system.

The vulnerability is a bypass of the previously addressed CVE-2023-39662, and a proof of concept has
been demonstrated that can create a file on the system by exploiting the flaw.

## Impact
By exploiting this vulnerability, an attacker can execute arbitrary code on the affected system,
leading to a complete compromise of confidentiality, integrity, and availability. This can result in
the attacker gaining full control of the system, stealing sensitive data, modifying or deleting
critical information, and disrupting the availability of the affected application or service.

## Mitigation
Users of the `llama_index` package are advised to upgrade to the fixed version [INSERT FIXED
VERSION] or later, which addresses this vulnerability. Additionally, it is recommended to review the
use of the `safe_eval` function and implement proper input validation to mitigate the risk of code
injection attacks.

## References
- CWE-94: Improper Control of Generation of Code ('Code Injection')
- CWE-20: Improper Input Validation
- CWE-502: Deserialization of Untrusted Data
- CVSS 3.1 Calculator: [INSERT LINK]

## Credit
This vulnerability was discovered and reported by [INSERT RESEARCHER NAME].

## Acknowledgments
The GitHub Security team would like to thank the researcher who discovered and reported this
vulnerability.
==============================================================================================
```

aws
security

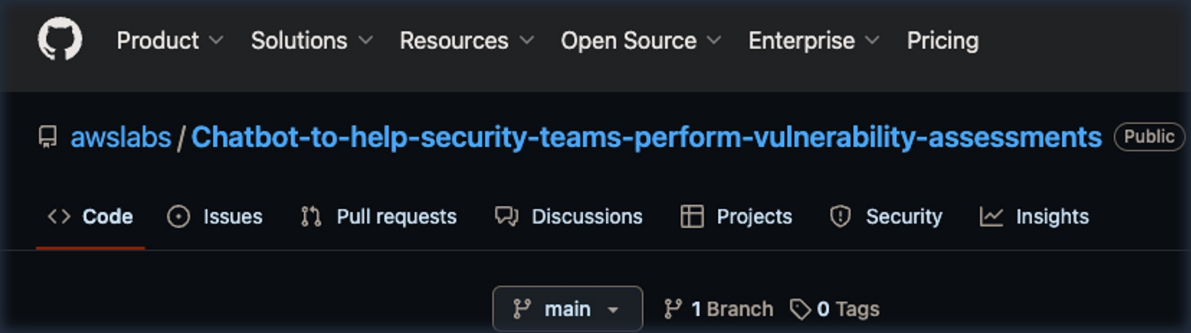
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Open-Sourcing and Community Engagement



Open source at AWS

Since its inception, Amazon Web Services (AWS) has been the best place for customers to build and run open source software in the cloud. AWS is proud to support open source projects, foundations, and partners. We believe that open source is good for everyone and we are committed to bringing the value of open source to our customers, and the operational excellence of AWS to open source communities.



awslabs / Chatbot-to-help-security-teams-perform-vulnerability-assessments

aws security

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Conclusion

Project Link:

- GitHub for CNA Guru project: https://github.com/awslabs/Chatbot-to-help-security-teams-perform-vulnerability-assessments

aws security

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

We look forward to hearing from you soon!

hackerone.com/aws_vdp | aws-security@amazon.com | aws.amazon.com/security/vulnerability-reporting