**Sri Lanka Institute of Information Technology**

**IE2062 - Web Security**

**Final Assignment**

**Bug Bounty Report 04**

**Name:** Peiris W.S.S.N

**Registration Number:** IT23227286

Contents

[**1.** **Introduction** 3](#_Toc197097282)

[**2.** **Reconnaissance** 3](#_Toc197097283)

[**3.** **Vulnerability** 5](#_Toc197097284)

[**4.** **Vulnerability description** 5](#_Toc197097285)

[**5.** **Affected Components** 5](#_Toc197097286)

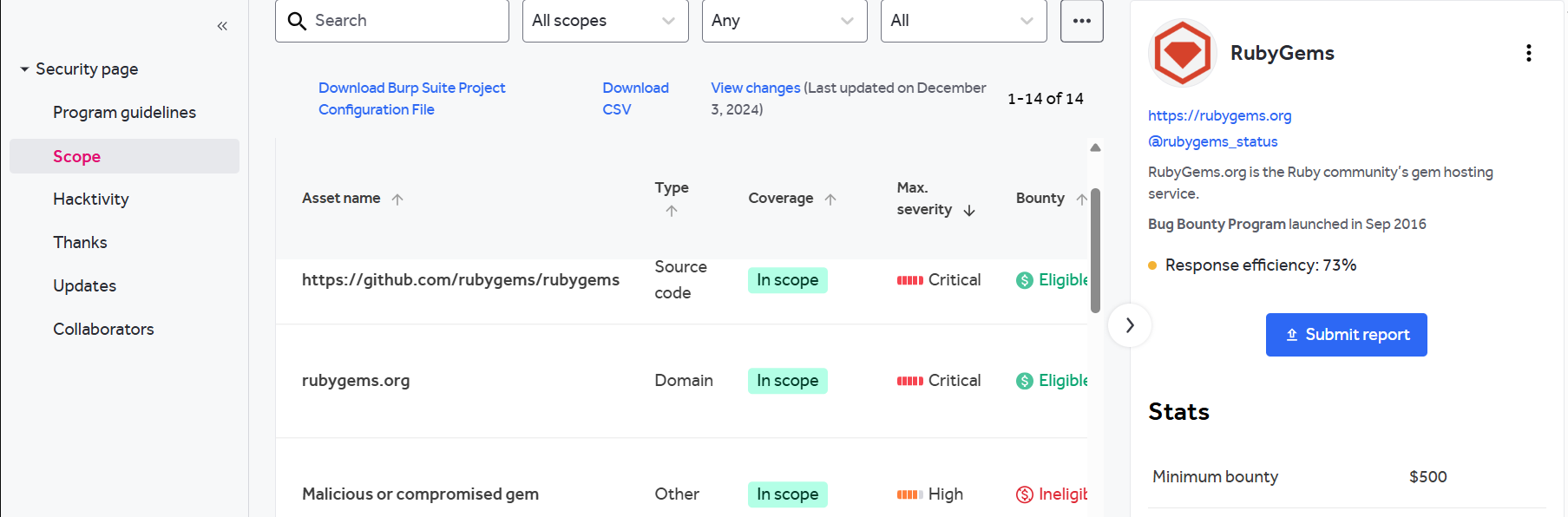
[**6.** **Impact Assessment** 6](#_Toc197097287)

[**7.** **Steps to reproduce** 6](#_Toc197097288)

[**8.** **Proof of concept** 7](#_Toc197097289)

[**9.** **Proposed mitigation or fix** 8](#_Toc197097290)

# **Introduction**

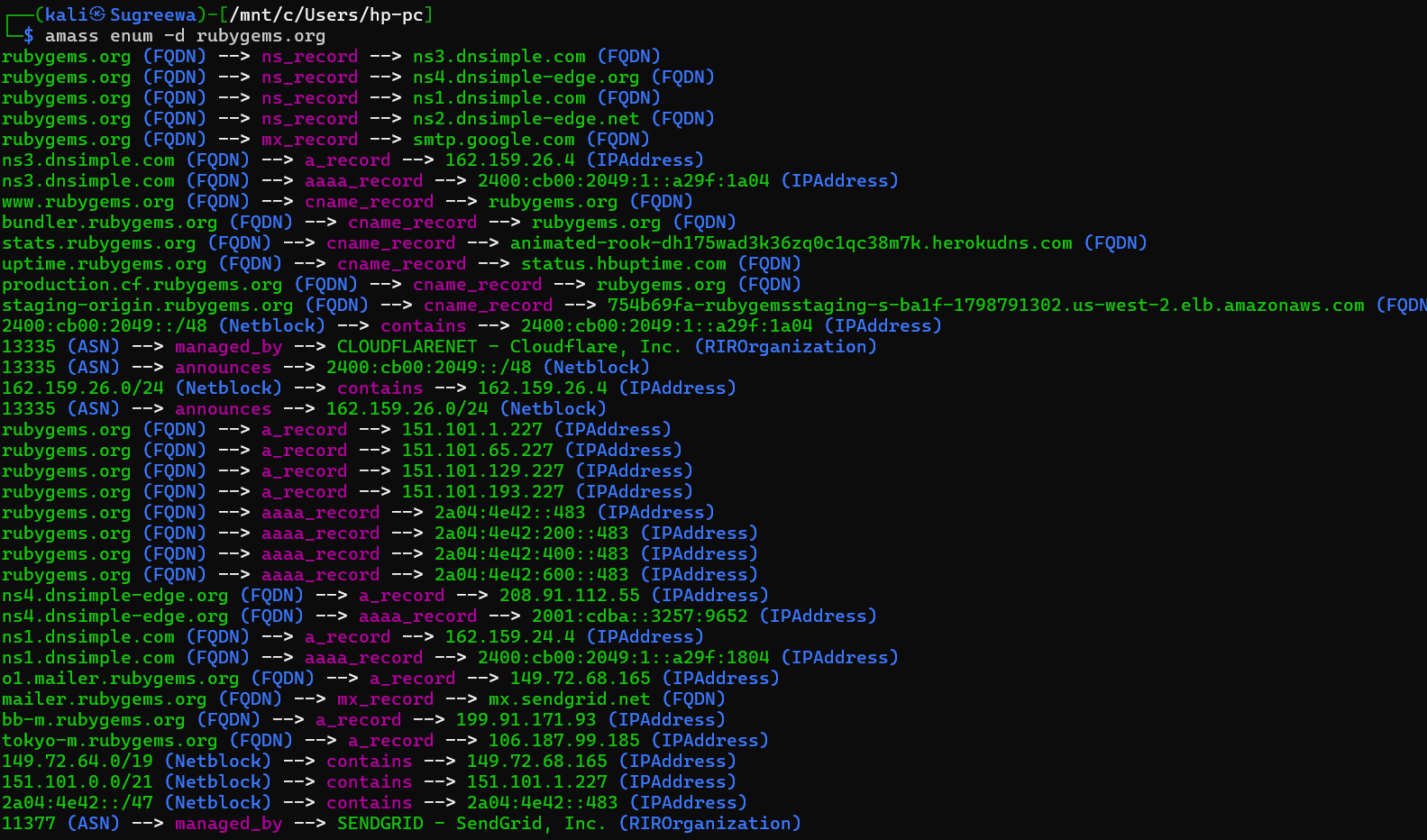


**Website:** <https://rubygems.org/>

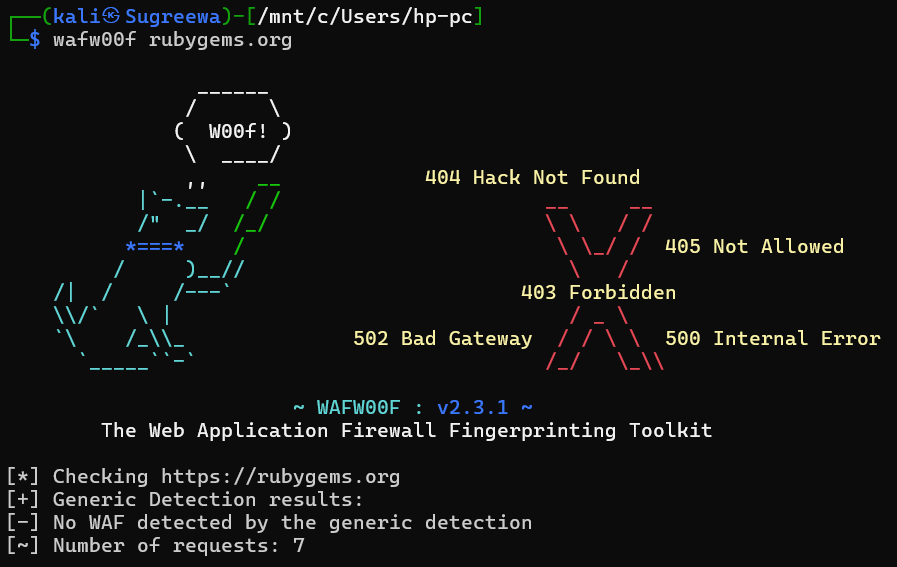
**Listed by:** RubyGems

# **Reconnaissance**

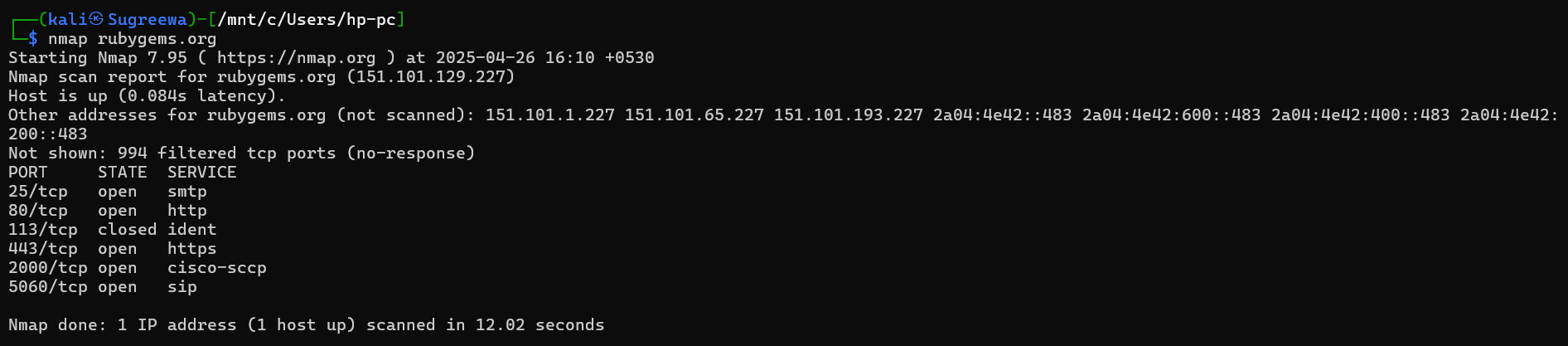
* **Subdomain enumeration using Amass**

****

* **Firewall Detection**

****

* **Nmap Scan**

****

# **Vulnerability**

* **CSP: style-src unsafe-inline**

****

# **Vulnerability description**

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files

1. **Affected Components**

* **Component:** HTTP Response Header → Content-Security-Policy
* **Directive Affected:** style-src
* **Policy:** style-src 'self' 'unsafe-inline' <https://fonts.googleapis.com>;
* **Impacted Pages:** All responses where this CSP header is set.

# **Impact Assessment**

* **Risk Level:** Medium to High (depending on other security controls in place)
* **Potential Impacts:**
  + Allows inline style injection (e.g., via HTML attributes like style="...")
  + May aid in **style-based XSS** or **UI redressing attacks**
  + Reduces defense-in-depth protections against content injection

# **Steps to reproduce**

* Use ZAP or browser DevTools to inspect the HTTP response headers from the target site.



* Identify the Content-Security-Policy header.

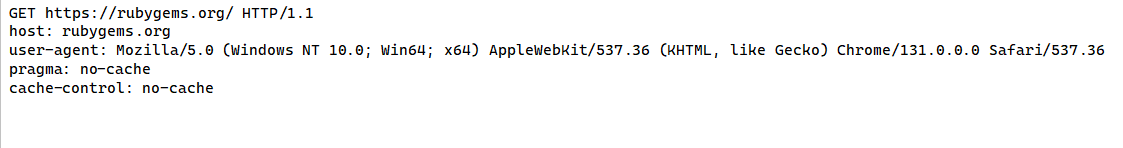


* Confirm that the style-src directive includes 'unsafe-inline'.

<p style="color: red; font-weight: bold;">This is a test of inline styles.</p>

* Optionally, test whether inline styles are accepted by injecting a simple HTML snippet with a style attribute.

# **Proof of concept**





# **Proposed mitigation or fix**

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.