Segurança no Debian: Como as falhas são corrigidas na prática

Moisés Sousa e Fernando Guisso

A palestra foi baseada na “A segurança do Debian ministrada pelo Samuel Henrique na MiniDebConf Maceió 2025:

https://www.youtube.com/watch?v=oxMU0pakk4s&t=1416s

https://samueloph.dev/about/

DebianDay
/// edição Natal - RN

Quem somos?

Moisés Sousa

Desenvolvedor web com atuação em projetos nacionais e internacionais em diversas stacks, atualmente focado em node e frontend. Mantenedor do projeto open-source “Querido Diário”. Graduado em TI pela UFRN e sou usuário de linux profissionalmente e pessoalmente há 8 anos.

https://www.linkedin.com/in/moisesmsa

https://github.com/MoisesMsa

Fernando Guisso

Trabalha com segurança de aplicações, com foco em desenvolvimento seguro, automação de testes e análise de vulnerabilidades. Participa ativamente de comunidades de tecnologia, como OWASP e SunsecRN, e gosta de compartilhar conhecimento em eventos e iniciativas open source. Nas horas vagas, está aprendendo bateria e tentando montar uma banda de um homem só.

https://www.linkedin.com/in/fernandoguisso

https://github.com/fguisso

DebianDay
/// edição Natal - RN

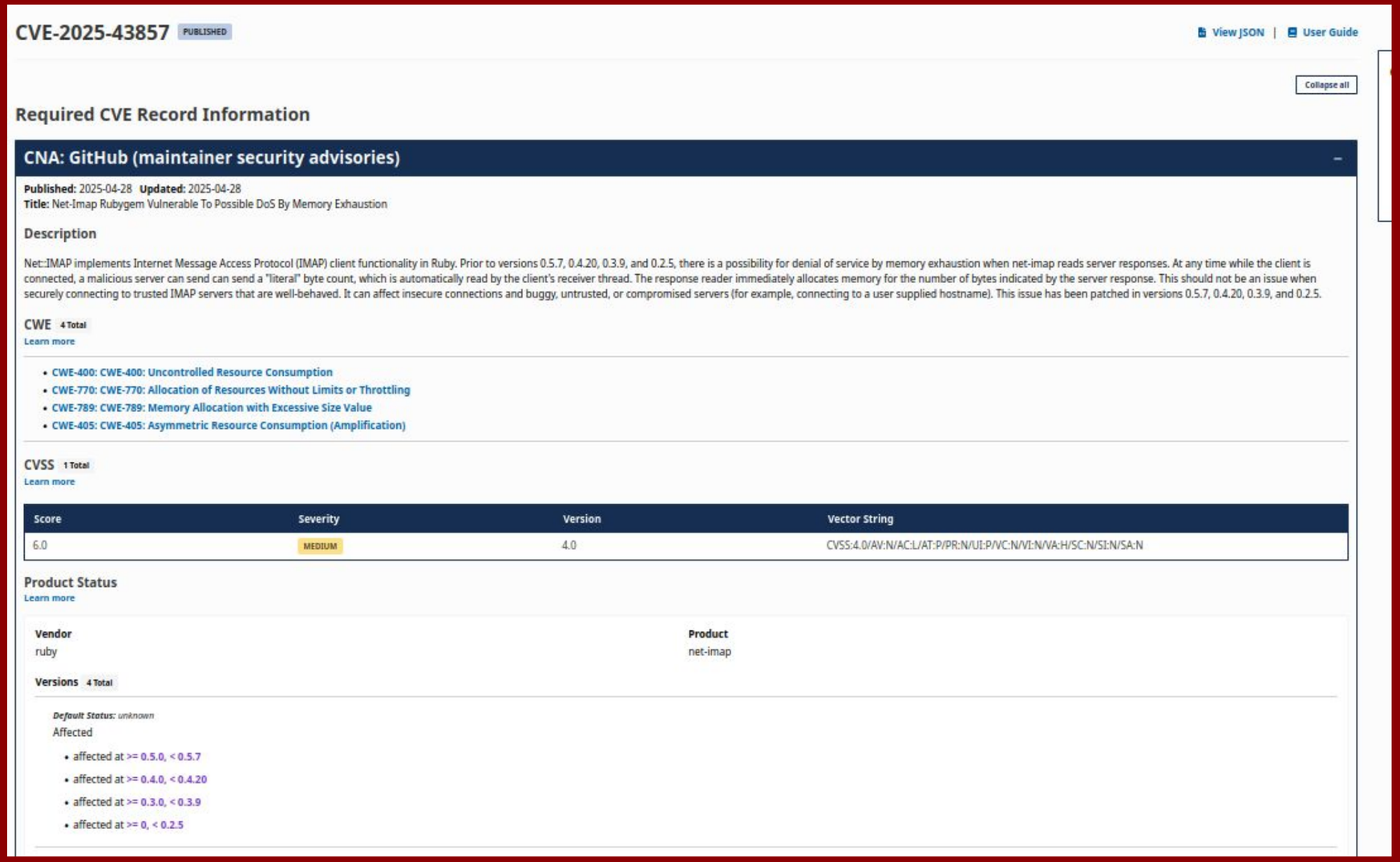
Visão geral

- Definir o que são CVEs
- Entender o funcionamento do suporte de segurança do Debian
- Entender como é feita a análise e correção de vulnerabilidades
- Entender como as correções chegam até o usuário final
- Dicas de investigação e boas práticas de segurança

O que são CVEs

CVE-YYYY-NNNN

- Common Vulnerabilities and Exposures
- Um ID universal como o seu CPF, por exemplo
- CVE.org coordenado pelo MITRE (EUA)
- Bancos de dados públicos (ex: NVD/NIST)
- CVSS: pontuação da severidade de 0 a 10
- CWE: categorização do tipo de falha (ex: buffer overflow)
- **Qualquer pessoa pode reportar uma CVE**



CVE-2025-43857 PUBLISHED

View JSON | User Guide

Collapse all

Required CVE Record Information

CNA: GitHub (maintainer security advisories)

Published: 2025-04-28 **Updated:** 2025-04-28
Title: Net-Imap Rubygem Vulnerable To Possible DoS By Memory Exhaustion

Description

Net::IMAP implements Internet Message Access Protocol (IMAP) client functionality in Ruby. Prior to versions 0.5.7, 0.4.20, 0.3.9, and 0.2.5, there is a possibility for denial of service by memory exhaustion when net-imap reads server responses. At any time while the client is connected, a malicious server can send can send a "literal" byte count, which is automatically read by the client's receiver thread. The response reader immediately allocates memory for the number of bytes indicated by the server response. This should not be an issue when securely connecting to trusted IMAP servers that are well-behaved. It can affect insecure connections and buggy, untrusted, or compromised servers (for example, connecting to a user supplied hostname). This issue has been patched in versions 0.5.7, 0.4.20, 0.3.9, and 0.2.5.

CWE 4 Total

Learn more

- CWE-400: CWE-400: Uncontrolled Resource Consumption
- CWE-770: CWE-770: Allocation of Resources Without Limits or Throttling
- CWE-789: CWE-789: Memory Allocation with Excessive Size Value
- CWE-405: CWE-405: Asymmetric Resource Consumption (Amplification)

CVSS 1 Total

Learn more

Score	Severity	Version	Vector String
6.0	MEDIUM	4.0	CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:P/VC:N/VI:N/VA:H/SC:N/SI:N/SA:N

Product Status

Learn more

Vendor
ruby

Product
net-imap

Versions 4 Total

Default Status: unknown
Affected

- affected at >= 0.5.0, < 0.5.7
- affected at >= 0.4.0, < 0.4.20
- affected at >= 0.3.0, < 0.3.9
- affected at >= 0, < 0.2.5

DebianDay
/// edição Natal - RN

https://www.cve.org/CVERecord?id=CVE-2025-43857

Information on source package ruby3.3

debian

ruby3.3 in the Package Tracking System | ruby3.3 in the Bug Tracking System | ruby3.3 source code | ruby3.3 in the testing migration checker

Available versions

Release	Version
trixie	3.3.8-2
forky	3.3.8-2
sid	3.3.8-2

Open issues

Bug	trixie	forky	sid	Description
CVE-2025-43857	vulnerable	vulnerable	vulnerable	Net::IMAP implements Internet Message Access Protocol (IMAP) client fu ...
CVE-2025-24294	vulnerable	vulnerable	vulnerable	The attack vector is a potential Denial of Service (DoS). The vulnerab ...

Resolved issues

Bug	Description
CVE-2025-27221	In the URI gem before 1.0.3 for Ruby, the URI handling methods (URI.jo ...
CVE-2025-27220	In the CGI gem before 0.4.2 for Ruby, a Regular Expression Denial of S ...
CVE-2025-27219	In the CGI gem before 0.4.2 for Ruby, the CGI::Cookie.parse method in ...
CVE-2025-25186	Net::IMAP implements Internet Message Access Protocol (IMAP) client fu ...
CVE-2025-0306	A vulnerability was found in Ruby. The Ruby interpreter is vulnerable ...
CVE-2024-49761	REXML is an XML toolkit for Ruby. The REXML gem before 3.3.9 has a ReD ...
CVE-2024-43398	REXML is an XML toolkit for Ruby. The REXML gem before 3.3.6 has a DoS ...
CVE-2024-41946	REXML is an XML toolkit for Ruby. The REXML gem 3.3.2 has a DoS vulner ...
CVE-2024-41123	REXML is an XML toolkit for Ruby. The REXML gem before 3.3.2 has some ...
CVE-2024-39908	REXML is an XML toolkit for Ruby. The REXML gem before 3.3.1 has some ...

Search for package or bug name: Go Reporting problems

Home Debian Security Source (Git)

DebianDay
/// edição Natal - RN

https://security-tracker.debian.org/tracker/source-package/ruby3.3

O que o Debian promete?

- 3 anos de suporte oficial (Debian Security Team)
- +2 anos via LTS (Long Term Support)
- +5 anos via ELTS (Extended LTS, pela Freexian)
- Total: até 10 anos de segurança
- Exemplo: Debian 8 (de 2015) ainda suportado via ELTS

Onde surgem as vulnerabilidades?

CVE (MITRE) – https://cve.mitre.org

- O padrão mais adotado globalmente para catalogar vulnerabilidades

Advisories por linguagem:

- GitHub Security Advisories (GHSA) – https://github.com/advisories
- RustSec (Rust) – https://rustsec.org/advisories/
- Go Vulnerability Database (Go) – https://vuln.go.dev

Sistemas nacionais de vulnerabilidades:

- JVN (Japão) – https://jvn.jp
- CNNVD (China) – http://www.cnnvd.org.cn
- RU-CERT (Rússia) – https://ructf.org
- EU-Vuln (UE) – sem uma base central pública, iniciativas como ENISA

Onde surgem as vulnerabilidades?

⚠️ **Nem toda falha tem CVE:**

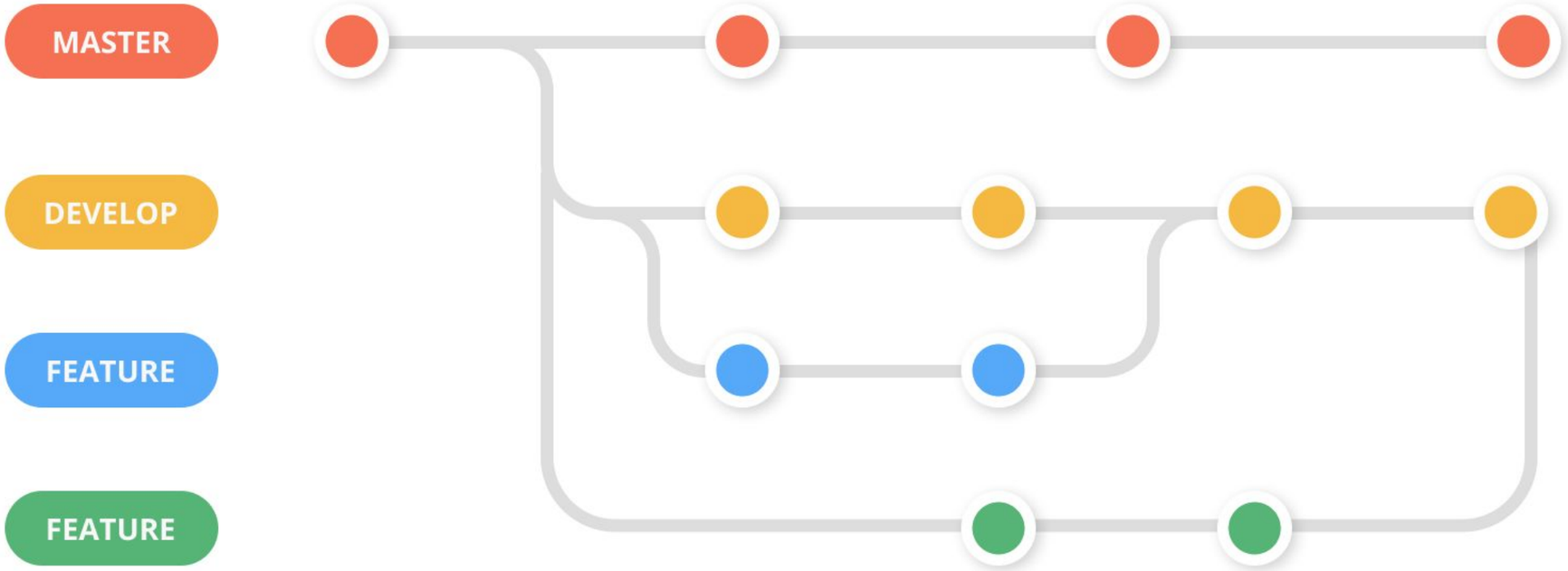
Algumas surgem primeiro no GitHub, fóruns ou advisories internos

🛠️ O Debian importa de todas essas fontes e avalia manualmente se o sistema é afetado

Como o Debian analiza vulnerabilidades?

- Coleta contínua de dados (CVE/Mitre, Github Advisory, RustSec, etc.)
- Investigação manual:
  - Entendimento da falha
  - Localização do "breaking commit"
  - Determinação de versões afetadas
- Análise pública no Security Tracker

DebianDay
/// edição Natal - RN



DebianDay
/// edição Natal - RN

Advisories (DSA/DLA/ELA)

Advisory é uma notificação pública emitida pelo time de segurança do Debian para alertar sobre uma ou mais falhas de segurança que foram corrigidas.Advisory é uma notificação pública emitida pelo time de segurança do Debian para alertar sobre uma ou mais falhas de segurança que foram corrigidas.

- CVEs graves podem vir sob embargo(zero-day)
- Correção coordenada antes da divulgação pública
- Exemplo: Backdoor no xz (2024)
- Debian, Fedora e outros corrigiram antes da publicação

DebianDay
/// edição Natal - RN

Como as atualizações chegam

- Correções graves: via repositório `stable-security`
  - Já está configurado por padrão `apt/source.list`
- Correções leves: via point releases `stable-updates` (a cada ~3 meses)
- Repositórios opcionais:
  - `proposed-updates` (instala antecipada com riscos)
- `apt install unattended-upgrades` atualização automática para estar sempre seguro, aceita configurações extras.

DebianDay
/// edição Natal - RN

Por que algumas CVEs não são corrigidas?

- Risco da correção ser maior que a falha
- Falta de mantenedor/disponibilidade
- A falha não é explorável na prática
- Solução requer remover uma feature
- Está no código-fonte, mas não é usado no build do Debian

Está no código-fonte, mas não é usado no build do Debian

- Build flags diferentes
  - O Debian compila os pacotes com parâmetros específicos (./configure, cmake, make, etc.).
  - Às vezes, features vulneráveis são desativadas por padrão no build do Debian.
- Código morto (dead code)
  - A vulnerabilidade está em uma função, módulo ou caminho de execução que não é chamado por nenhuma parte do programa.
- Feature desativada em tempo de execução
  - Mesmo que o código tenha sido compilado, o Debian configura o software para não usar aquela parte vulnerável.

Boas Práticas

- Use Debian ou derivadas sérias (atenção com distribuições menores)
- Faça atualizações frequentes
- Entenda quando reiniciar (pacote crítico ≠ serviço reiniciado)
- Pacotes não oficiais = maior risco
- Remova o que não usa

Conclusão

- A maioria das CVEs **não é crítica**
- Segurança se faz com vigilância contínua
- Use as ferramentas e os espaços da comunidade Debian para acompanhar (Security Tracker!)

Participe, contribua e compartilhe o conhecimento

DebianDay
/// edição Natal - RN

Obrigado

DebianDay
/// edição Natal - RN