KAZ HACK STAN

2025

SEPTEMBER 17–19
ALMATY

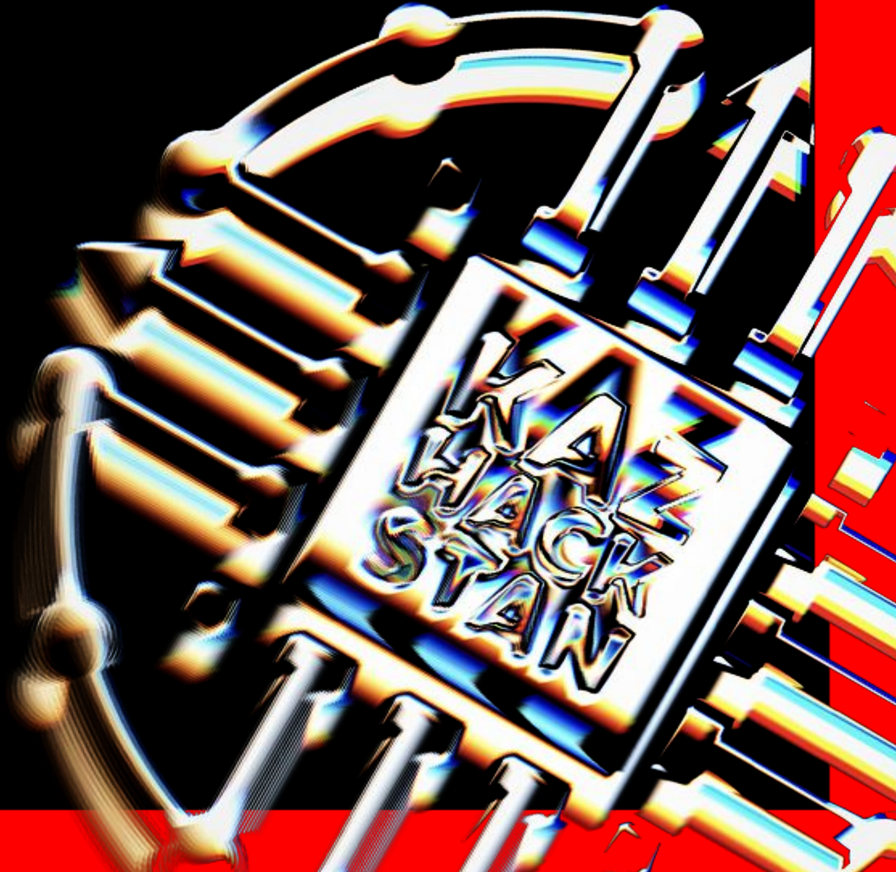
TSARKA · DIGITAL & SPACE MINISTRY · FREEDOM HOLDING CORP. · TREND MICRO · astana hub

HACKDAY

Hacking with AI SASTs: From Hype to Hands-On Results

SPEAKER

Joshua Rogers



ABOUT ME

HACKDAY

Joshua Rogers

Executive Senior Principal Lead Head of Security Architectural Engineering

15 years in the scene. experience in troublemaking, asking the difficult/wrong questions, hacking, exploit dev, sec engineering, programming, incident responding, incident creating, problem solving, trend-setting, trolling, bartending, travelling, … and so on.

Taking the ethical out of ethical hacking, and putting the offensive into offensive security.

Currently: Freelance work + AppSec Engineer/Security Researcher for a Crypto Company.

TL;DR; Why you should care

- Multiple AI-native SASTs on the market today.
- They work *extremely* well.
- They find real vulnerabilities and logic bugs in minutes.
- They can “think”/”reason” about business logic issues.
- They match developer intent with actual code.
- They aren’t based on static rule-sets and queries.
- They have low false positive rates.
- They’re cheap (for now).

HACKDAY

Context

Trying to find some magic.

AI SAST: A Working Definition

A tool whose input is source code, and whose output is (any of) the following:

1. Vulnerabilities that create security risk.
2. Malicious code (intentional or not).
3. Major bugs that impact stability/security.

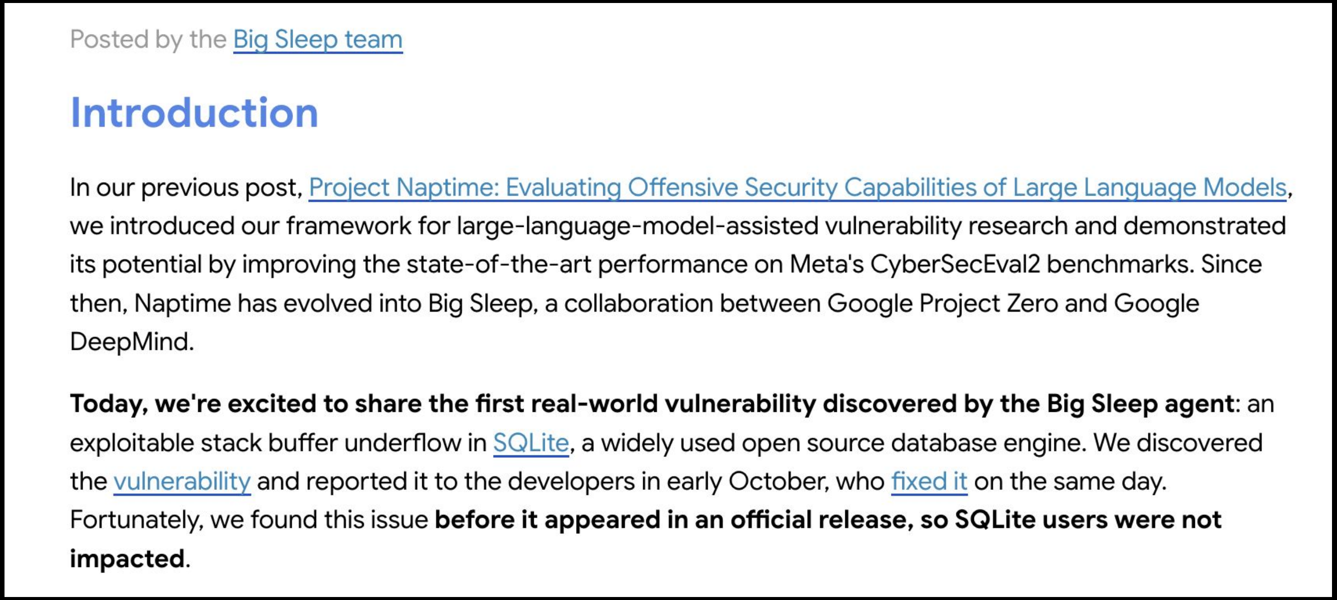
Anything else is just a bonus.

HACKDAY

Google's Magic Bug Hunter: Big Sleep

- 2024: Begins finding vulns.
- 2025: Continues to find criticals in ffmpeg, V8, Ghostscript, JavaScriptCore, imagemagick, etc.

Public Bug Tracker: https://goo.gle/bigsleep



Posted by the Big Sleep team

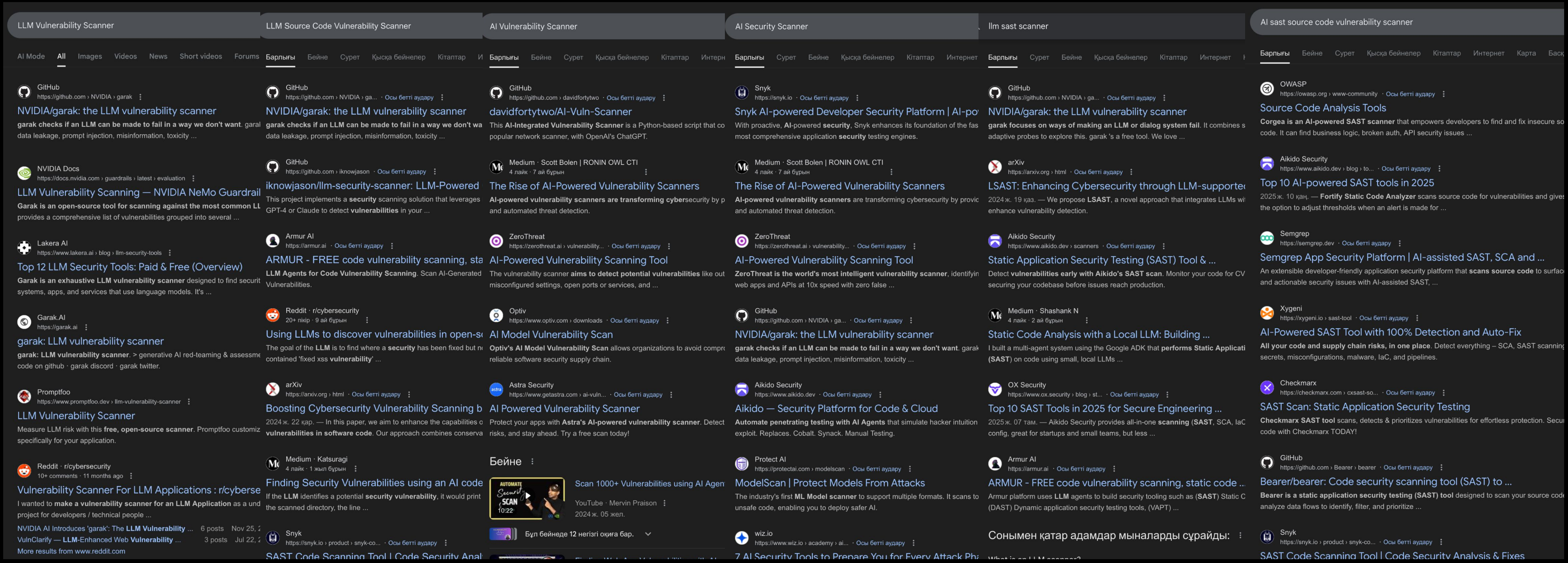
Introduction

In our previous post, Project Naptime: Evaluating Offensive Security Capabilities of Large Language Models, we introduced our framework for large-language-model-assisted vulnerability research and demonstrated its potential by improving the state-of-the-art performance on Meta's CyberSecEval2 benchmarks. Since then, Naptime has evolved into Big Sleep, a collaboration between Google Project Zero and Google DeepMind.

Today, we're excited to share the first real-world vulnerability discovered by the Big Sleep agent: an exploitable stack buffer underflow in SQLite, a widely used open source database engine. We discovered the vulnerability and reported it to the developers in early October, who fixed it on the same day. Fortunately, we found this issue **before it appeared in an official release, so SQLite users were not impacted**.

HACKDAY

How do I find somebody to give money to?


LLM Vulnerability Scanner
AI Mode All Images Videos News Short videos Forums
LLM Source Code Vulnerability Scanner
AI Vulnerability Scanner
AI Security Scanner
llm sast scanner
AI sast source code vulnerability scanner
Барлығы Бейне Сурет Қысқа бейнелер Кітаптар Интернет Карта

LLM Vulnerability Scanner

GitHub · https://github.com › NVIDIA › garak
NVIDIA/garak: the LLM vulnerability scanner
garak checks if an LLM can be made to fail in a way we don't want. garal data leakage, prompt injection, misinformation, toxicity ...

NVIDIA Docs · https://docs.nvidia.com › guardrails › latest › evaluation
LLM Vulnerability Scanning — NVIDIA NeMo Guardrail
Garak is an open-source tool for scanning against the most common LL provides a comprehensive list of vulnerabilities grouped into several ...

Lakera AI · https://www.lakera.ai › blog › llm-security-tools
Top 12 LLM Security Tools: Paid & Free (Overview)
Garak is an exhaustive LLM vulnerability scanner designed to find securit systems, apps, and services that use language models. It's ...

Garak.AI · https://garak.ai
garak: LLM vulnerability scanner
garak: LLM vulnerability scanner. > generative AI red-teaming & assessme code on github · garak discord · garak twitter.

Promptfoo · https://www.promptfoo.dev › llm-vulnerability-scanner
LLM Vulnerability Scanner
Measure LLM risk with this **free, open-source scanner**. Promptfoo customiz specifically for your application.

Reddit · r/cybersecurity · 10+ comments · 11 months ago
Vulnerability Scanner For LLM Applications : r/cyberse
I wanted to **make a vulnerability scanner for an LLM Application** as a und project for developers / technical people ...
NVIDIA AI Introduces 'garak': The **LLM Vulnerability** ... 6 posts Nov 25, 2
VulnClarify — **LLM**-Enhanced Web **Vulnerability** ... 3 posts Jul 22, 2
More results from www.reddit.com

LLM Source Code Vulnerability Scanner

GitHub · https://github.com › NVIDIA › ga... · Осы бетті аудару
NVIDIA/garak: the LLM vulnerability scanner
garak checks if an LLM can be made to fail in a way we don't wa data leakage, prompt injection, misinformation, toxicity ...

GitHub · https://github.com › iknowjason · Осы бетті аудару
iknowjason/llm-security-scanner: LLM-Powered
This project implements a **security** scanning solution that leverages GPT-4 or Claude to detect **vulnerabilities** in your ...

Armur AI · https://armur.ai · Осы бетті аудару
ARMUR - FREE code vulnerability scanning, sta
LLM Agents for Code Vulnerability Scanning. Scan AI-Generated Vulnerabilities.

Reddit · r/cybersecurity · 20+ пікір · 9 ай бұрын
Using LLMs to discover vulnerabilities in open-s
The goal of the **LLM** is to find where a **security** has been fixed but n contained 'fixed xss **vulnerability**' ...

arXiv · https://arxiv.org › html · Осы бетті аудару
Boosting Cybersecurity Vulnerability Scanning b
2024 ж. 22 қар. — In this paper, we aim to enhance the capabilities o **vulnerabilities in software code**. Our approach combines conserva

Medium · Katsuragi · 4 лайк · 1 жыл бұрын
Finding Security Vulnerabilities using an AI code
If the **LLM** identifies a potential **security vulnerability**, it would print the scanned directory, the line ...

Snyk · https://snyk.io › product › snyk-co... · Осы бетті аудару
SAST Code Scanning Tool | Code Security Anal

AI Vulnerability Scanner

GitHub · https://github.com › davidfortytwo · Осы бетті аудару
davidfortytwo/AI-Vuln-Scanner
This **AI-Integrated Vulnerability Scanner** is a Python-based script that co popular network scanner, with OpenAI's ChatGPT.

Medium · Scott Bolen | RONIN OWL CTI · 4 лайк · 7 ай бұрын
The Rise of AI-Powered Vulnerability Scanners
AI-powered vulnerability scanners are transforming cybersecurity by p and automated threat detection.

ZeroThreat · https://zerothreat.ai › vulnerability... · Осы бетті аудару
AI-Powered Vulnerability Scanning Tool
The vulnerability scanner **aims to detect potential vulnerabilities** like out misconfigured settings, open ports or services, and ...

Optiv · https://www.optiv.com › downloads · Осы бетті аудару
AI Model Vulnerability Scan
Optiv's AI Model Vulnerability Scan allows organizations to avoid compr reliable software security supply chain.

Astra Security · https://www.getastra.com › ai-vuln... · Осы бетті аудару
AI Powered Vulnerability Scanner
Protect your apps with **Astra's AI-powered vulnerability scanner**. Detect risks, and stay ahead. Try a free scan today!

Бейне

AUTOMATE Security SCAN 10:22
Scan 1000+ Vulnerabilities using AI Agen
YouTube · Mervin Praison
2024 ж. 05 жел.
Бұл бейнеде 12 негізгі оқиға бар.

AI Security Scanner

Snyk · https://snyk.io · Осы бетті аудару
Snyk AI-powered Developer Security Platform | AI-po
With proactive, **AI**-powered **security**, Snyk enhances its foundation of the fas most comprehensive application **security** testing engines.

Medium · Scott Bolen | RONIN OWL CTI · 4 лайк · 7 ай бұрын
The Rise of AI-Powered Vulnerability Scanners
AI-powered vulnerability scanners are transforming cybersecurity by provic and automated threat detection.

ZeroThreat · https://zerothreat.ai › vulnerability... · Осы бетті аудару
AI-Powered Vulnerability Scanning Tool
ZeroThreat is the world's most intelligent vulnerability scanner, identifyin web apps and APIs at 10x speed with zero false ...

GitHub · https://github.com › NVIDIA › ga... · Осы бетті аудару
NVIDIA/garak: the LLM vulnerability scanner
garak checks if an LLM can be made to fail in a way we don't want. garak data leakage, prompt injection, misinformation, toxicity ...

Aikido Security · https://www.aikido.dev · Осы бетті аудару
Aikido — Security Platform for Code & Cloud
Automate penetrating testing with AI Agents that simulate hacker intuition exploit. Replaces. Cobalt. Synack. Manual Testing.

Protect AI · https://protectai.com › modelscan · Осы бетті аудару
ModelScan | Protect Models From Attacks
The industry's first **ML Model scanner** to support multiple formats. It scans to unsafe code, enabling you to deploy safer AI.

wiz.io · https://www.wiz.io › academy › ai... · Осы бетті аудару
7 AI Security Tools to Prepare You for Every Attack Pha

llm sast scanner

GitHub · https://github.com › NVIDIA › ga... · Осы бетті аудару
NVIDIA/garak: the LLM vulnerability scanner
garak focuses on ways of making an LLM or dialog system fail. It combines s adaptive probes to explore this. garak 's a free tool. We love ...

arXiv · https://arxiv.org › html · Осы бетті аудару
LSAST: Enhancing Cybersecurity through LLM-supporte
2024 ж. 19 қаз. — We propose **LSAST**, a novel approach that integrates LLMs wi enhance vulnerability detection.

Aikido Security · https://www.aikido.dev › scanners · Осы бетті аудару
Static Application Security Testing (SAST) Tool & ...
Detect **vulnerabilities early with Aikido's SAST scan**. Monitor your code for CV securing your codebase before issues reach production.

Medium · Shashank N · 4 лайк · 2 ай бұрын
Static Code Analysis with a Local LLM: Building ...
I built a multi-agent system using the Google ADK that **performs Static Applicati** (**SAST**) on code using small, local LLMs ...

OX Security · https://www.ox.security › blog › st... · Осы бетті аудару
Top 10 SAST Tools in 2025 for Secure Engineering ...
2025 ж. 07 там. — Aikido Security provides all-in-one **scanning** (**SAST**, SCA, IaC config, great for startups and small teams, but less ...

Armur AI · https://armur.ai · Осы бетті аудару
ARMUR - FREE code vulnerability scanning, static code ...
Armur platform uses **LLM** agents to build security tooling such as (**SAST**) Static C (DAST) Dynamic application security testing tools, (VAPT) ...

Сонымен қатар адамдар мыналарды сұрайды:

AI sast source code vulnerability scanner

OWASP · https://owasp.org › www-community · Осы бетті аудару
Source Code Analysis Tools
Corgea is an AI-powered SAST scanner that empowers developers to find and fix insecure so code. It can find business logic, broken auth, API security issues ...

Aikido Security · https://www.aikido.dev › blog › to... · Осы бетті аудару
Top 10 AI-powered SAST tools in 2025
2025 ж. 10 қаң. — **Fortify Static Code Analyzer** scans source code for vulnerabilities and gives the option to adjust thresholds when an alert is made for ...

Semgrep · https://semgrep.dev · Осы бетті аудару
Semgrep App Security Platform | AI-assisted SAST, SCA and ...
An extensible developer-friendly application security platform that **scans source code** to surfac and actionable security issues with AI-assisted SAST, ...

Xygeni · https://xygeni.io › sast-tool · Осы бетті аудару
AI-Powered SAST Tool with 100% Detection and Auto-Fix
All your code and supply chain risks, in one place. Detect everything – SCA, SAST scanning secrets, misconfigurations, malware, IaC, and pipelines.

Checkmarx · https://checkmarx.com › cxsast-so... · Осы бетті аудару
SAST Scan: Static Application Security Testing
Checkmarx SAST tool scans, detects & prioritizes vulnerabilities for effortless protection. Secu code with Checkmarx TODAY!

GitHub · https://github.com › Bearer › bearer · Осы бетті аудару
Bearer/bearer: Code security scanning tool (SAST) to ...
Bearer is a static application security testing (SAST) tool designed to scan your source code analyze data flows to identify, filter, and prioritize ...

Snyk · https://snyk.io › product › snyk-co... · Осы бетті аудару
SAST Code Scanning Tool | Code Security Analysis & Fixes

Landscape

What’s out there & common capabilities.

Current Vendors

- ***Almanax***: "The AI Security Engineer"
- ***Amplify Security***: "Develop Secure Software Confidently"
- ***Corgea***: "Smarter AppSec, built with AI"
- ***DryRun Security***: "Codebase Risk Averted w/ Contextual Security Analysis"
- ***Gecko Security***: "AI Security Engineer"
- ***ZeroPath***: "AI-Native SAST & AppSec Platform"

Common Functionality

Code Scans

- GH/GL/… & File Upload.
- Full Code Scans.
- Branch Scans.
- PR/MR Scans.
- "Taint/Flow" Analysis.
- F/P Detection.
- Custom Policies/Rules.
- Scheduled, Automatic, Recurring, On-Demand.

CI/CD Hooks

- Hooks to scan changes.
- Blocking/non-blocking.
- Bot responses in PRs.
- Alerts to Slack, et al.
- IDE Plugins (some).

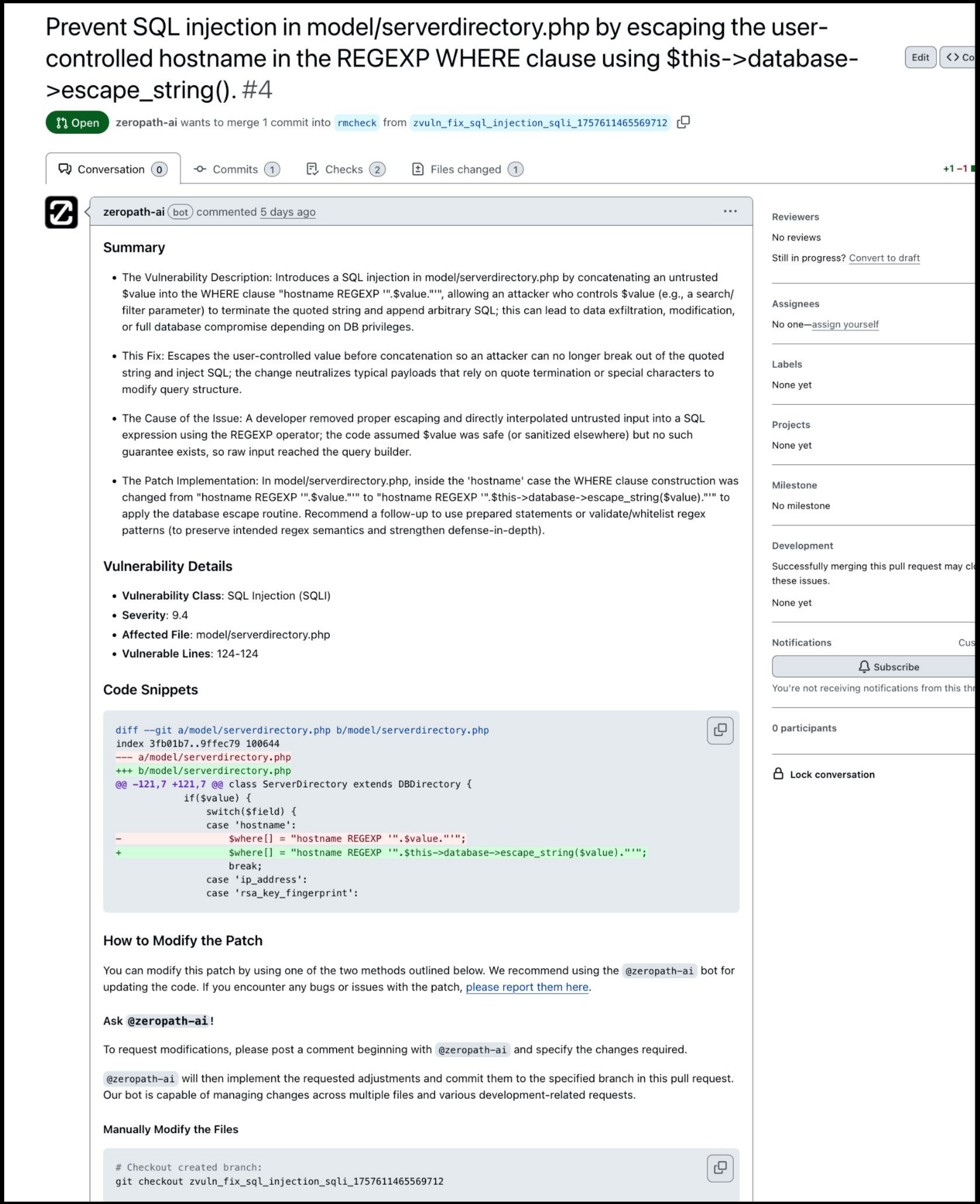
Auto-Patching

- Remediation Guidance.
- Offer suggested patches.
- Submit patches as PRs.

HACKDAY

Bonus Stuff

- SSO Support.
- No-Retention Policies.
- Audit Trails & Logging.
- RBAC Permissions.
- Cross-Domain Identity Management (SCIM).
- Reporting Dashboards.
- Reporting Exports.
- Dependency Scanning.
- APIs.
- Integrations (other SASTs).
- Report Exporting (CSV, SARIF, PDF).
- Policy-As-Scan.



Prevent SQL injection in model/serverdirectory.php by escaping the user-controlled hostname in the REGEXP WHERE clause using $this->database->escape_string(). #4

zeropath-ai wants to merge 1 commit into `rmcheck` from `zvuln_fix_sql_injection_sqli_1757611465569712`

zeropath-ai (bot) commented 5 days ago

Summary

- The Vulnerability Description: Introduces a SQL injection in model/serverdirectory.php by concatenating an untrusted $value into the WHERE clause "hostname REGEXP '".$value."'", allowing an attacker who controls $value (e.g., a search/filter parameter) to terminate the quoted string and append arbitrary SQL; this can lead to data exfiltration, modification, or full database compromise depending on DB privileges.
- This Fix: Escapes the user-controlled value before concatenation so an attacker can no longer break out of the quoted string and inject SQL; the change neutralizes typical payloads that rely on quote termination or special characters to modify query structure.
- The Cause of the Issue: A developer removed proper escaping and directly interpolated untrusted input into a SQL expression using the REGEXP operator; the code assumed $value was safe (or sanitized elsewhere) but no such guarantee exists, so raw input reached the query builder.
- The Patch Implementation: In model/serverdirectory.php, inside the 'hostname' case the WHERE clause construction was changed from "hostname REGEXP '".$value."'" to "hostname REGEXP '".$this->database->escape_string($value)."'" to apply the database escape routine. Recommend a follow-up to use prepared statements or validate/whitelist regex patterns (to preserve intended regex semantics and strengthen defense-in-depth).

Vulnerability Details

- **Vulnerability Class**: SQL Injection (SQLI)
- **Severity**: 9.4
- **Affected File**: model/serverdirectory.php
- **Vulnerable Lines**: 124-124

Code Snippets

```
diff --git a/model/serverdirectory.php b/model/serverdirectory.php
index 3fb01b7..9ffec79 100644
--- a/model/serverdirectory.php
+++ b/model/serverdirectory.php
@@ -121,7 +121,7 @@ class ServerDirectory extends DBDirectory {
            if($value) {
                switch($field) {
                case 'hostname':
-                   $where[] = "hostname REGEXP '".$value."'";
+                   $where[] = "hostname REGEXP '".$this->database->escape_string($value)."'";
                    break;
                case 'ip_address':
                case 'rsa_key_fingerprint':
```

How to Modify the Patch

You can modify this patch by using one of the two methods outlined below. We recommend using the `@zeropath-ai` bot for updating the code. If you encounter any bugs or issues with the patch, please report them here.

Ask `@zeropath-ai`!

To request modifications, please post a comment beginning with `@zeropath-ai` and specify the changes required.

`@zeropath-ai` will then implement the requested adjustments and commit them to the specified branch in this pull request. Our bot is capable of managing changes across multiple files and various development-related requests.

Manually Modify the Files

```
# Checkout created branch:
git checkout zvuln_fix_sql_injection_sqli_1757611465569712
```

Reviewers
No reviews
Still in progress? Convert to draft

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may cl
these issues.
None yet

Notifications
Cus

You're not receiving notifications from this thr

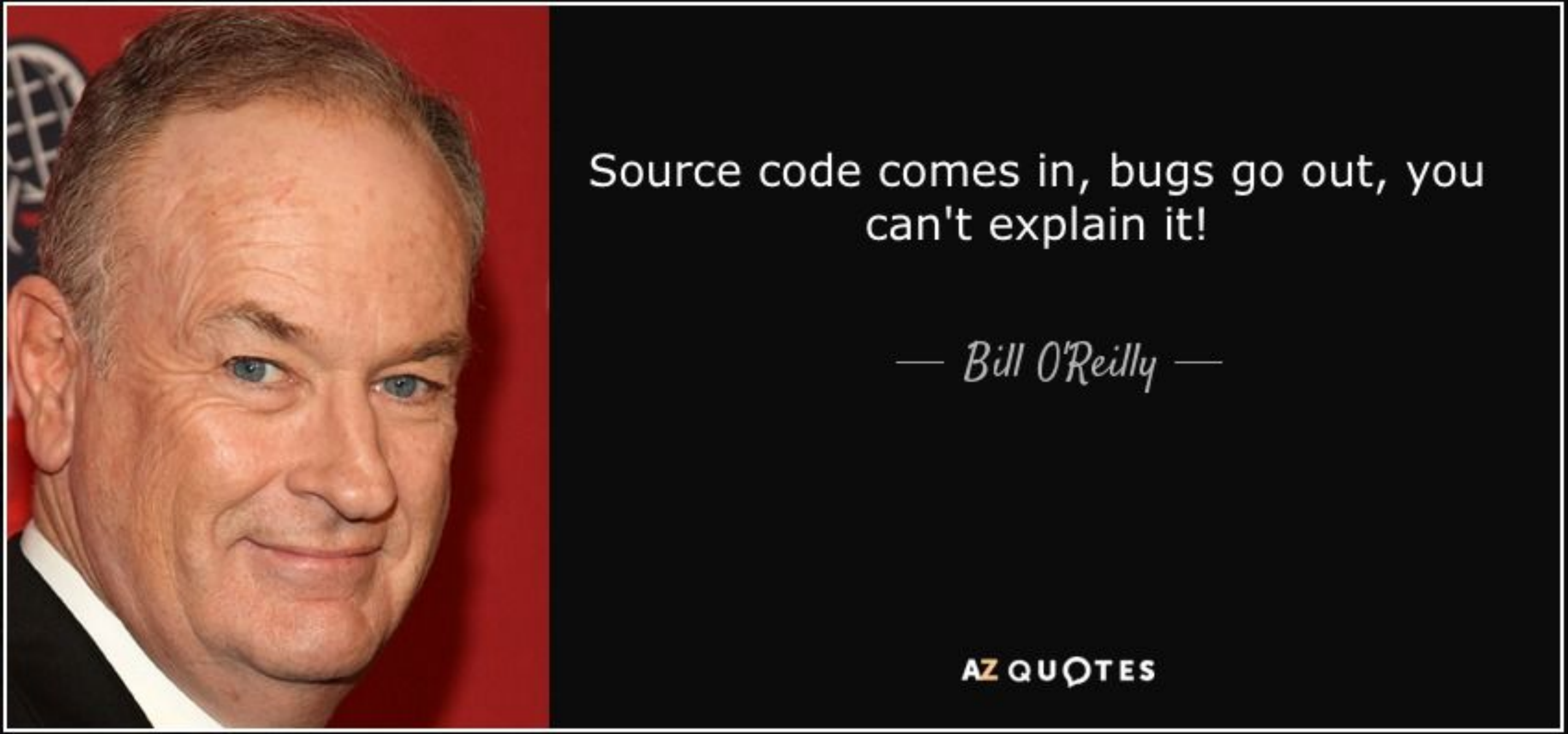
0 participants

HACKDAY

How They Work

Maybe.

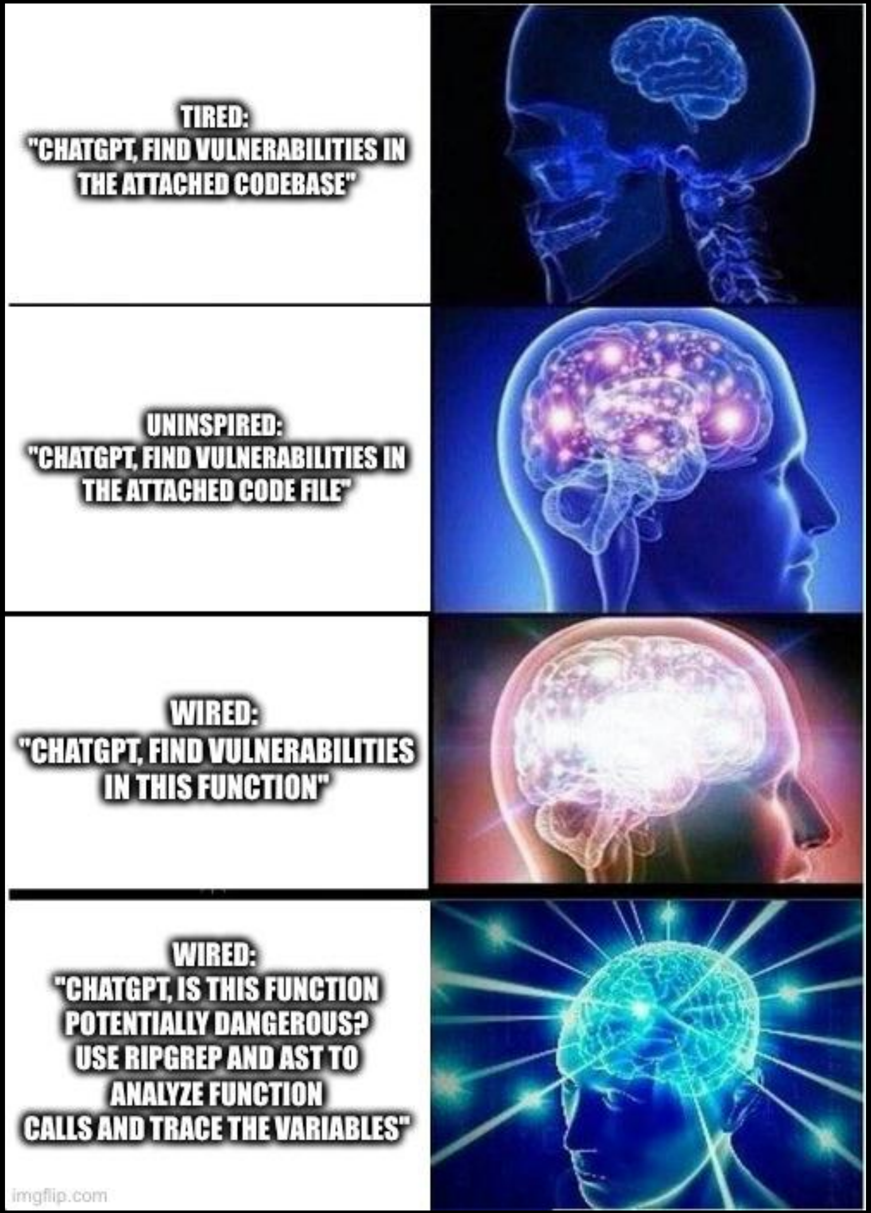
HACKDAY

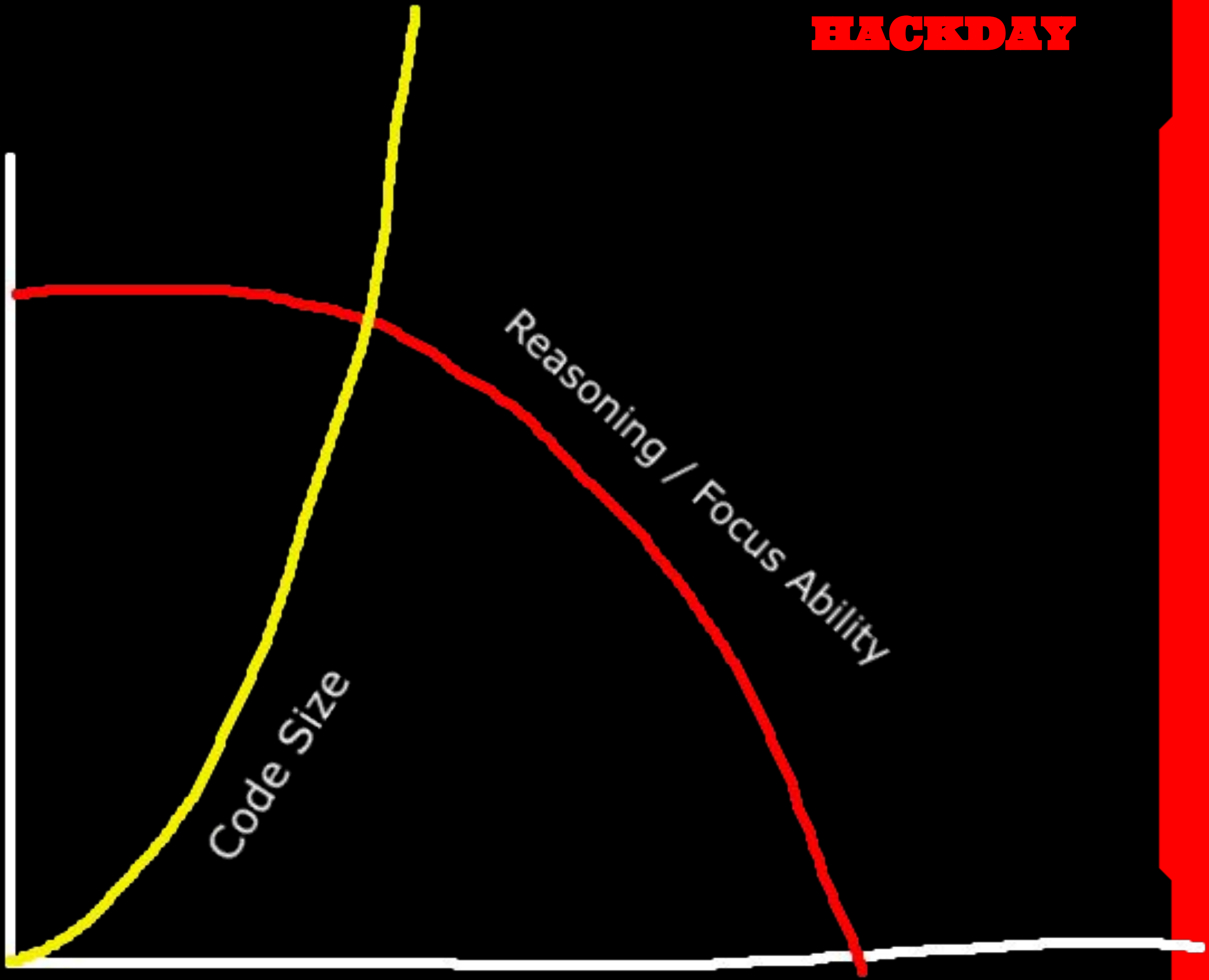


Source code comes in, bugs go out, you can't explain it!

— Bill O'Reilly —

AZ QUOTES


TIRED:
"CHATGPT, FIND VULNERABILITIES IN THE ATTACHED CODEBASE"
UNINSPIRED:
"CHATGPT, FIND VULNERABILITIES IN THE ATTACHED CODE FILE"
WIRED:
"CHATGPT, FIND VULNERABILITIES IN THIS FUNCTION"
WIRED:
"CHATGPT, IS THIS FUNCTION POTENTIALLY DANGEROUS? USE RIPGREP AND AST TO ANALYZE FUNCTION CALLS AND TRACE THE VARIABLES"
imgflip.com


Code Size
Reasoning / Focus Ability

How They Work.. Maybe.

HACKDAY

Stage 1: Intake

- Code Retrieval.
- AST Generation.
- Indexing of Code.
- Context Enrichment.
- App Identification.
- Dependency Identification.
- Behavior Analysis.

Stage 2: Discovery

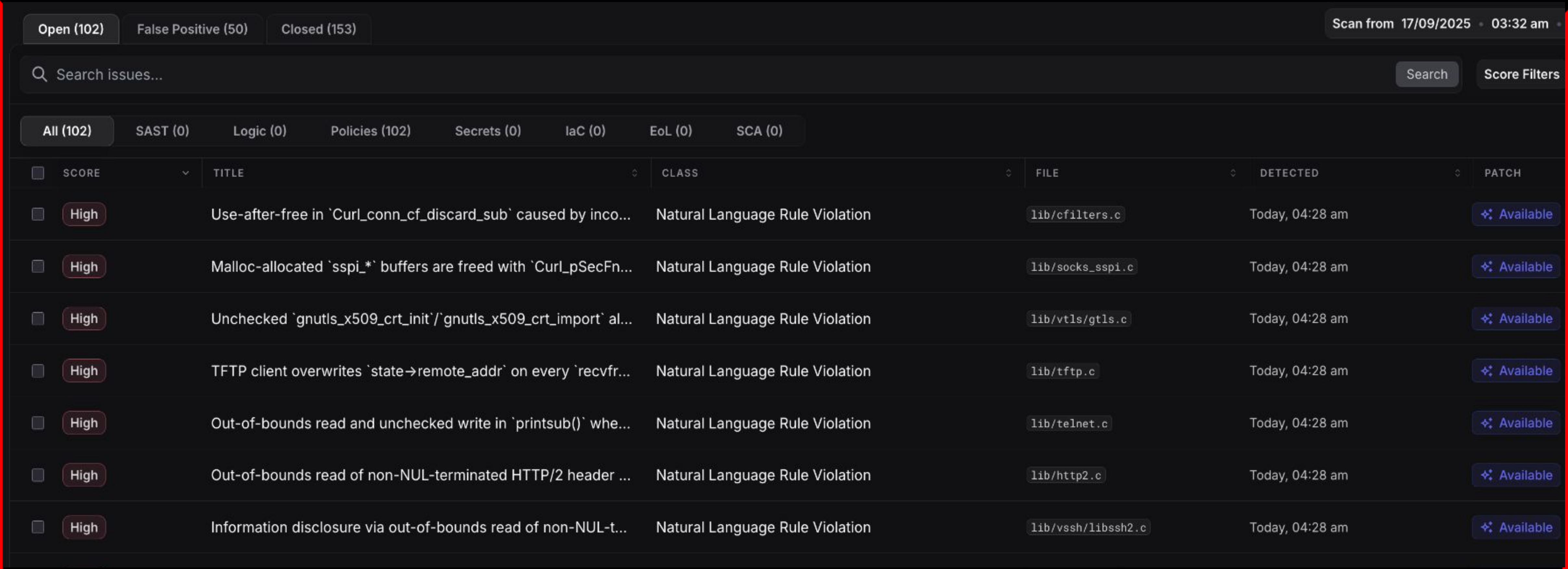
- Query LLM with data.
- Query with opengrep rules.
- Custom SASTs.
- Tools e.g. ripgrep.
- Function Analysis.
- Risky Behavior Analysis.
- Protection Checks.
- Contextual Checks.
- Authorization Checks.
- Usage Identification.
- Source/Sink Analysis.
- Custom “Rules”.

Stage 3: Validation

- False Positive Detection.
- (More) Context Retrieval.
- Taint Reasoning.
- Function-Level Prompts.
- Duplicate Detection.
- Reachability Analysis.
- Context Analysis.
- Trust Boundary Analysis.
- Severity Scoring.
- Patch Generation.

ZeroPath: Result Viewer

HACKDAY



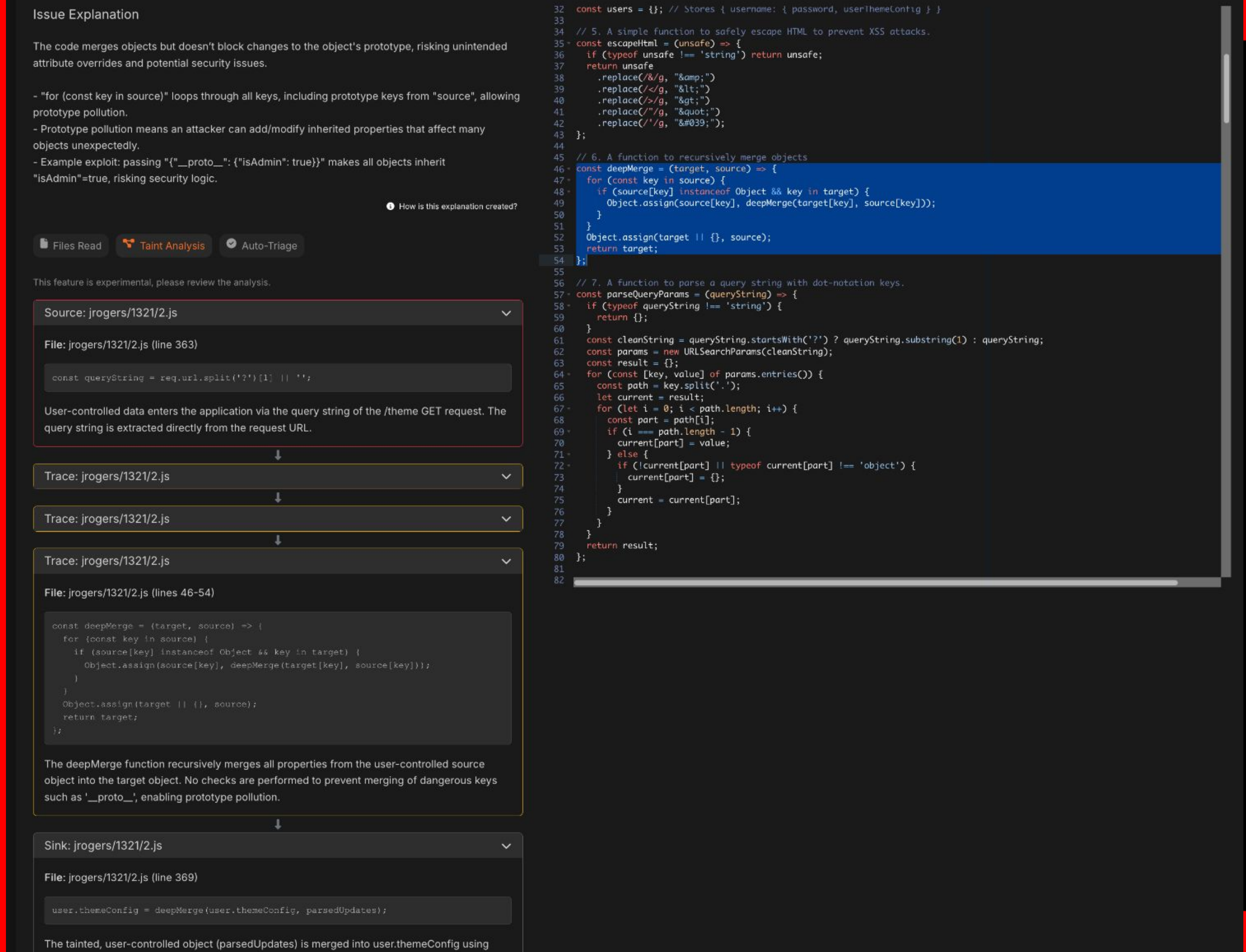
Open (102) | False Positive (50) | Closed (153)

Scan from 17/09/2025 • 03:32 am

Search issues...

All (102) | SAST (0) | Logic (0) | Policies (102) | Secrets (0) | IaC (0) | EoL (0) | SCA (0)

SCORE	TITLE	CLASS	FILE	DETECTED	PATCH
High	Use-after-free in `Curl_conn_cf_discard_sub` caused by inco...	Natural Language Rule Violation	`lib/cfilters.c`	Today, 04:28 am	Available
High	Malloc-allocated `sspi_*` buffers are freed with `Curl_pSecFn...	Natural Language Rule Violation	`lib/socks_sspi.c`	Today, 04:28 am	Available
High	Unchecked `gnutls_x509_crt_init`/`gnutls_x509_crt_import` al...	Natural Language Rule Violation	`lib/vtls/gtls.c`	Today, 04:28 am	Available
High	TFTP client overwrites `state->remote_addr` on every `recvfr...	Natural Language Rule Violation	`lib/tftp.c`	Today, 04:28 am	Available
High	Out-of-bounds read and unchecked write in `printsub()` whe...	Natural Language Rule Violation	`lib/telnet.c`	Today, 04:28 am	Available
High	Out-of-bounds read of non-NUL-terminated HTTP/2 header ...	Natural Language Rule Violation	`lib/http2.c`	Today, 04:28 am	Available
High	Information disclosure via out-of-bounds read of non-NUL-t...	Natural Language Rule Violation	`lib/vssh/libssh2.c`	Today, 04:28 am	Available



Issue Explanation

The code merges objects but doesn't block changes to the object's prototype, risking unintended attribute overrides and potential security issues.

- "for (const key in source)" loops through all keys, including prototype keys from "source", allowing prototype pollution.
- Prototype pollution means an attacker can add/modify inherited properties that affect many objects unexpectedly.
- Example exploit: passing "{"__proto__": {"isAdmin": true}}" makes all objects inherit "isAdmin"=true, risking security logic.

How is this explanation created?

Files Read | Taint Analysis | Auto-Triage

This feature is experimental, please review the analysis.

Source: jrogers/1321/2.js

File: jrogers/1321/2.js (line 363)

```
const queryString = req.url.split('?')[1] || '';
```

User-controlled data enters the application via the query string of the /theme GET request. The query string is extracted directly from the request URL.

Trace: jrogers/1321/2.js

Trace: jrogers/1321/2.js

Trace: jrogers/1321/2.js

File: jrogers/1321/2.js (lines 46-54)

```
const deepMerge = (target, source) => {
  for (const key in source) {
    if (source[key] instanceof Object && key in target) {
      Object.assign(source[key], deepMerge(target[key], source[key]));
    }
  }
  Object.assign(target || {}, source);
  return target;
};
```

The deepMerge function recursively merges all properties from the user-controlled source object into the target object. No checks are performed to prevent merging of dangerous keys such as '__proto__', enabling prototype pollution.

Sink: jrogers/1321/2.js

File: jrogers/1321/2.js (line 369)

```
user.themeConfig = deepMerge(user.themeConfig, parsedUpdates);
```

The tainted, user-controlled object (parsedUpdates) is merged into user.themeConfig using

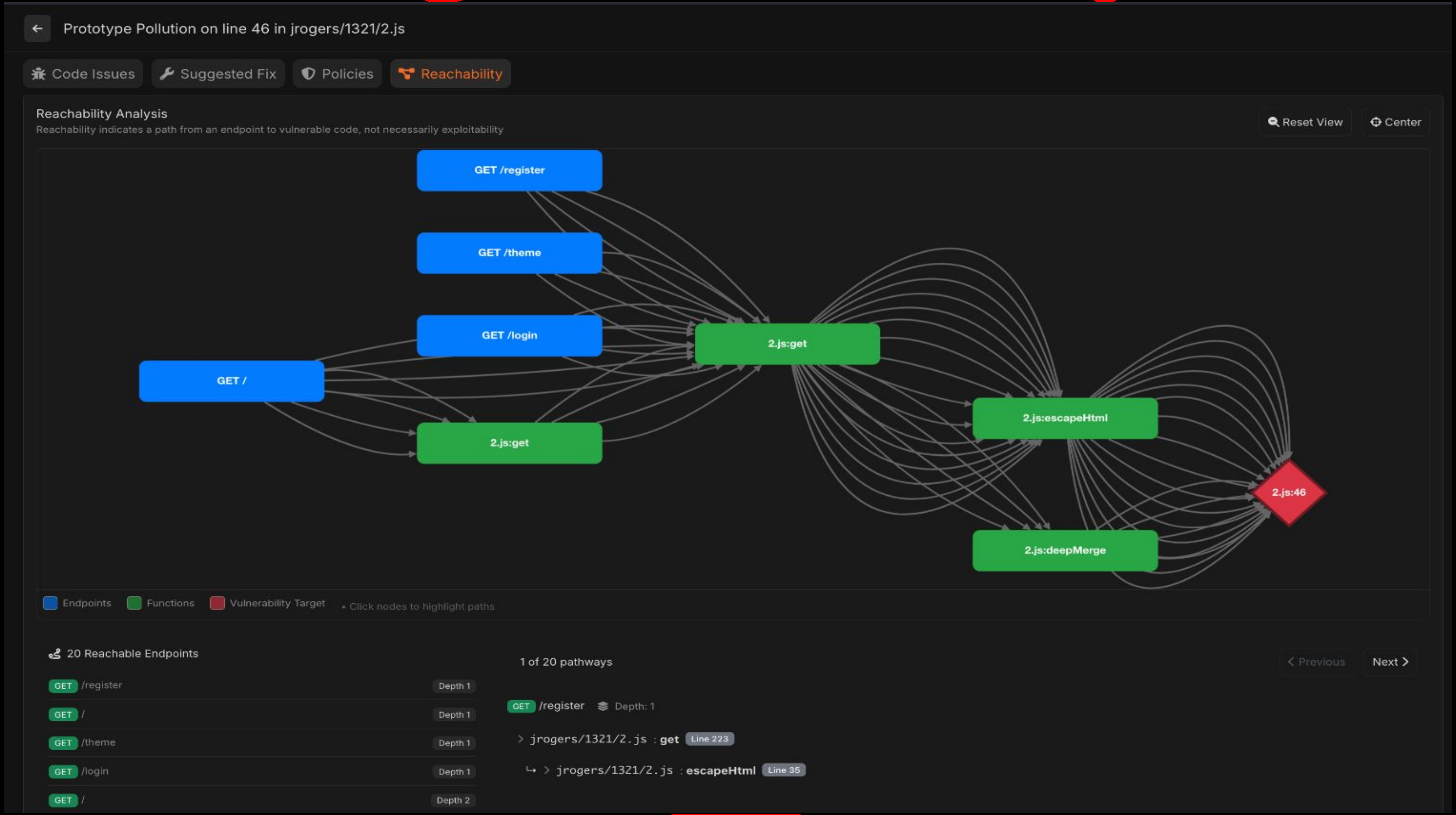
```
32 const users = {}; // Stores { username: { password, userThemeConfig } }
33
34 // 5. A simple function to safely escape HTML to prevent XSS attacks.
35 const escapeHtml = (unsafe) => {
36   if (typeof unsafe !== 'string') return unsafe;
37   return unsafe
38     .replace(/&/g, "&amp;")
39     .replace(/</g, "&lt;")
40     .replace(/>/g, "&gt;")
41     .replace(/"/g, "&quot;")
42     .replace(/'/g, "&#039;");
43 };
44
45 // 6. A function to recursively merge objects
46 const deepMerge = (target, source) => {
47   for (const key in source) {
48     if (source[key] instanceof Object && key in target) {
49       Object.assign(source[key], deepMerge(target[key], source[key]));
50     }
51   }
52   Object.assign(target || {}, source);
53   return target;
54 };
55
56 // 7. A function to parse a query string with dot-notation keys.
57 const parseQueryParams = (queryString) => {
58   if (typeof queryString !== 'string') {
59     return {};
60   }
61   const cleanString = queryString.startsWith('?') ? queryString.substring(1) : queryString;
62   const params = new URLSearchParams(cleanString);
63   const result = {};
64   for (const [key, value] of params.entries()) {
65     const path = key.split('.');
66     let current = result;
67     for (let i = 0; i < path.length; i++) {
68       const part = path[i];
69       if (i === path.length - 1) {
70         current[part] = value;
71       } else {
72         if (!current[part] || typeof current[part] !== 'object') {
73           current[part] = {};
74         }
75         current = current[part];
76       }
77     }
78   }
79   return result;
80 };
81
82
```

HACKDAY

Corgea: Issue Viewer

Corgea: Reachability

HACKDAY



Prototype Pollution on line 46 in jrogers/1321/2.js

Code Issues | Suggested Fix | Policies | Reachability

Reachability Analysis

Reachability indicates a path from an endpoint to vulnerable code, not necessarily exploitability

Reset View | Center

🟦 Endpoints 🟩 Functions 🟥 Vulnerability Target • Click nodes to highlight paths

20 Reachable Endpoints

GET /register	Depth 1
GET /	Depth 1
GET /theme	Depth 1
GET /login	Depth 1
GET /	Depth 2

1 of 20 pathways

< Previous | Next >

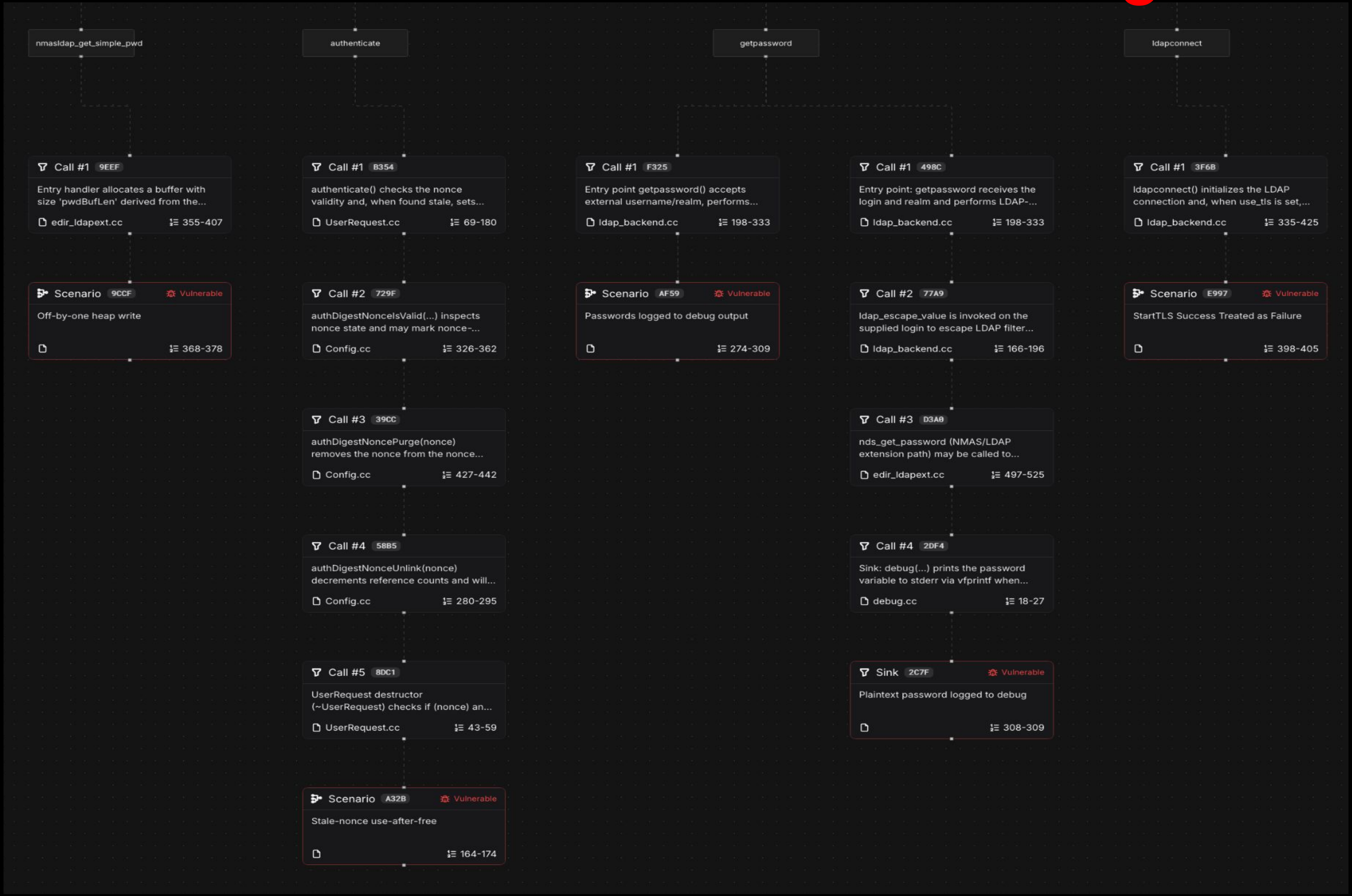
GET /register Depth: 1

> jrogers/1321/2.js : get Line 223

↳ > jrogers/1321/2.js : escapeHtml Line 35

ZeroPath: Reachability

HACKDAY



nmasldap_get_simple_pwd

Call #1 9EEF
Entry handler allocates a buffer with size 'pwdBufLen' derived from the...
edir_ldapext.cc 355-407

Scenario 9CCF Vulnerable
Off-by-one heap write
368-378

authenticate

Call #1 B354
authenticate() checks the nonce validity and, when found stale, sets...
UserRequest.cc 69-180

Call #2 729F
authDigestNonceIsValid(...) inspects nonce state and may mark nonce-...
Config.cc 326-362

Call #3 39CC
authDigestNoncePurge(nonce) removes the nonce from the nonce...
Config.cc 427-442

Call #4 58B5
authDigestNonceUnlink(nonce) decrements reference counts and will...
Config.cc 280-295

Call #5 8DC1
UserRequest destructor (~UserRequest) checks if (nonce) an...
UserRequest.cc 43-59

Scenario A32B Vulnerable
Stale-nonce use-after-free
164-174

getpassword

Call #1 F325
Entry point getpassword() accepts external username/realm, performs...
ldap_backend.cc 198-333

Scenario AF59 Vulnerable
Passwords logged to debug output
274-309

Call #1 498C
Entry point: getpassword receives the login and realm and performs LDAP-...
ldap_backend.cc 198-333

Call #2 77A9
ldap_escape_value is invoked on the supplied login to escape LDAP filter...
ldap_backend.cc 166-196

Call #3 D3A0
nds_get_password (NMAS/LDAP extension path) may be called to...
edir_ldapext.cc 497-525

Call #4 2DF4
Sink: debug(...) prints the password variable to stderr via vfprintf when...
debug.cc 18-27

Sink 2C7F Vulnerable
Plaintext password logged to debug
308-309

ldapconnect

Call #1 3F6B
ldapconnect() initializes the LDAP connection and, when use_tls is set,...
ldap_backend.cc 335-425

Scenario E997 Vulnerable
StartTLS Success Treated as Failure
398-405

Results

What went well; what didn’t.

HACKDAY

The Good

Top 3:

- ZeroPath.
- Corgea.
- Almanax.

Needs Improving:

- Gecko Security.
- Amplify Security.

Untested:

- DryRun Security.

Findings:

- Hundreds of real vulns.
- Excellent bug-hunting results.
- Serious logic/business bugs.
- Serious architectural issues.
- Esoteric/context-specific issues.
- High coverage even in neglected code.
- Completely broken functionality.
- RFC/Spec violations.
- Low(-ish) false positive rate.
- Low false negative rate.

Top Findings In:

- sudo.
- libwebm.
- Next.js.
- avahi
- wpa_supplicant
- curl
- squid (200+)

The Not So Good

HACKDAY

Findings:

- Some inconsistent critical findings (“*Just run it a second time*”).
- Malicious code detection didn’t work at all (except Almanax).
- False positive detection didn't always work so well.
- Deduplication gaps.
- Security drift: trivial issues labeled critical
- Difficult to understand report descriptions (sc
- Poor patch generation.
- Some poor interoperability understanding and ir
- Poor issue taxonomy.
- Some logic bugs missed.
- Prompt Injection (lol).
- Lots of UI bugs (but all comp

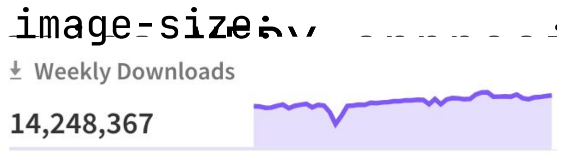
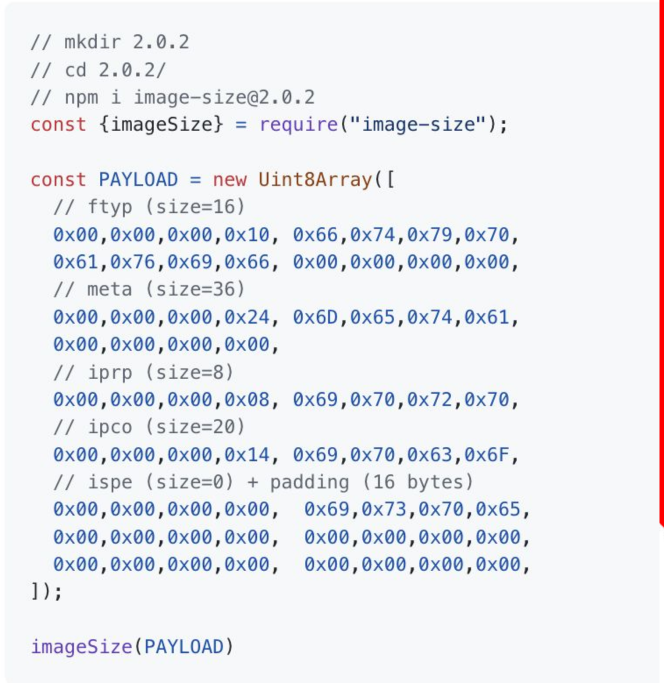


image-size:

Weekly Downloads

14,248,367



```
// mkdir 2.0.2
// cd 2.0.2/
// npm i image-size@2.0.2
const {imageSize} = require("image-size");

const PAYLOAD = new Uint8Array([
  // ftyp (size=16)
  0x00,0x00,0x00,0x10, 0x66,0x74,0x79,0x70,
  0x61,0x76,0x69,0x66, 0x00,0x00,0x00,0x00,
  // meta (size=36)
  0x00,0x00,0x00,0x24, 0x6D,0x65,0x74,0x61,
  0x00,0x00,0x00,0x00,
  // iprp (size=8)
  0x00,0x00,0x00,0x08, 0x69,0x70,0x72,0x70,
  // ipco (size=20)
  0x00,0x00,0x00,0x14, 0x69,0x70,0x63,0x6F,
  // ispe (size=0) + padding (16 bytes)
  0x00,0x00,0x00,0x00,  0x69,0x73,0x70,0x65,
  0x00,0x00,0x00,0x00,  0x00,0x00,0x00,0x00,
  0x00,0x00,0x00,0x00,  0x00,0x00,0x00,0x00,
]);

imageSize(PAYLOAD)
```

General Results

HACKDAY

Almanax

- Excellent single-function "obvious" results.
- Not so good at large/complicated code.
- Great at simple malicious code detection.
- Raw-bones solutions, not yet a mature product.

Corgea

- Discovered nearly all "test-case" issues.
- Discovered real vulns in big codebases.
- Tons of F/Ps.
- Malicious detection sucks.
- Excellent UI & reports.
- Tons of bugs in UI.
- PR reviews failed hard.

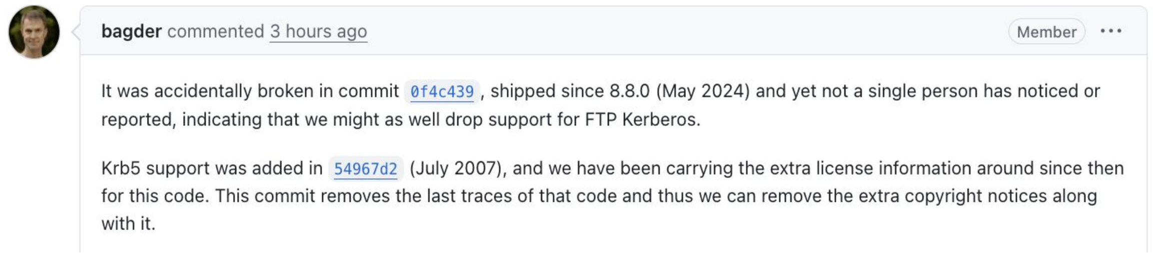
ZeroPath

- Discovered all "test-case" issues.
- Intimidatingly good bug and vuln findings.
- Excellent PR scanning.
- In-built issue chatbot.
- Even better with policies.
- Extremely slow UI.
- Complex issuedescriptions.

Funny Findings

HACKDAY

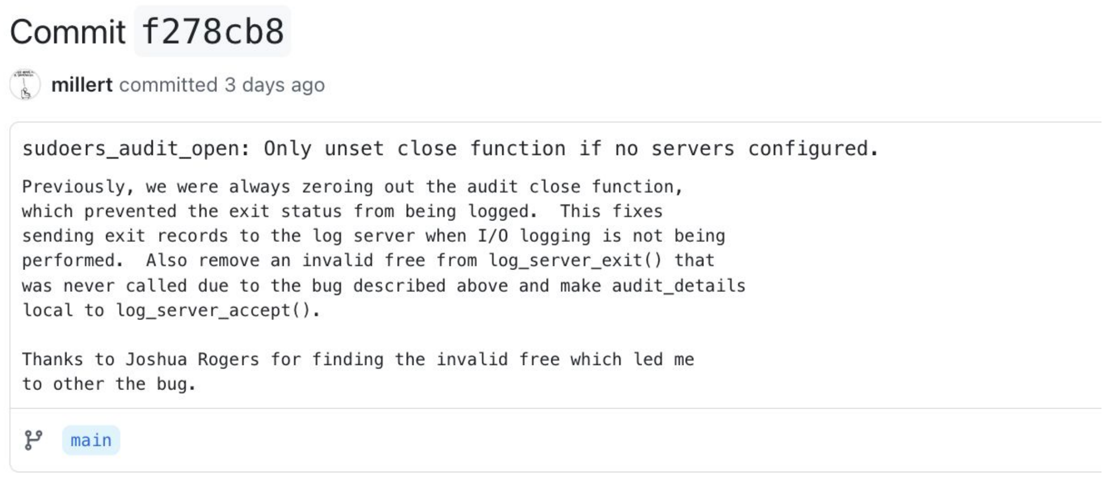
curl: vulnerability in .. broken code.


bagder commented 3 hours ago · Member

It was accidentally broken in commit 0f4c439, shipped since 8.8.0 (May 2024) and yet not a single person has noticed or reported, indicating that we might as well drop support for FTP Kerberos.

Krb5 support was added in 54967d2 (July 2007), and we have been carrying the extra license information around since then for this code. This commit removes the last traces of that code and thus we can remove the extra copyright notices along with it.

sudo: vulnerability in .. broken code.

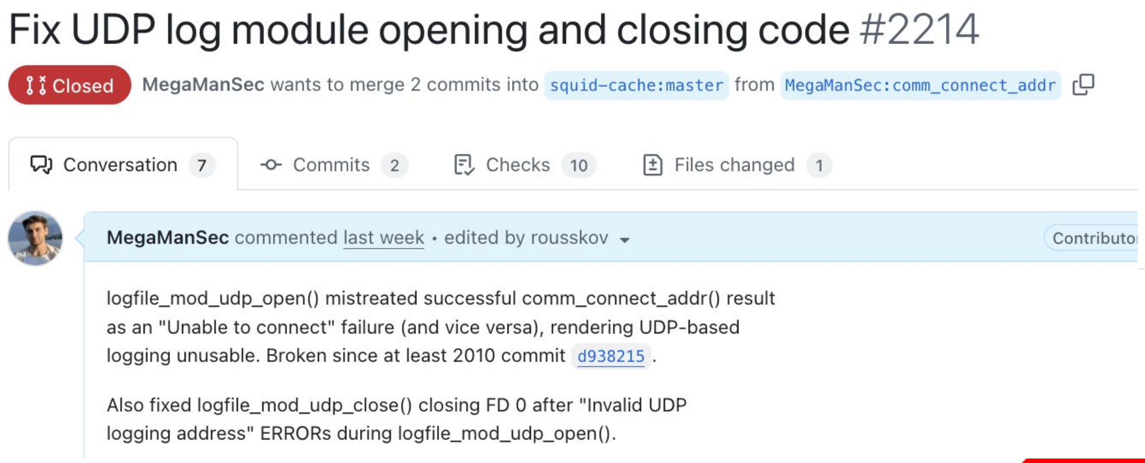

Commit f278cb8

millert committed 3 days ago

```
sudoers_audit_open: Only unset close function if no servers configured.

Previously, we were always zeroing out the audit close function,
which prevented the exit status from being logged.  This fixes
sending exit records to the log server when I/O logging is not being
performed.  Also remove an invalid free from log_server_exit() that
was never called due to the bug described above and make audit_details
local to log_server_accept().

Thanks to Joshua Rogers for finding the invalid free which led me
to other the bug.
```

main

squid: vulnerability in .. broken code.


Fix UDP log module opening and closing code #2214

Closed · MegaManSec wants to merge 2 commits into squid-cache:master from MegaManSec:comm_connect_addr

Conversation 7 · Commits 2 · Checks 10 · Files changed 1

MegaManSec commented last week • edited by rousskov · Contributor

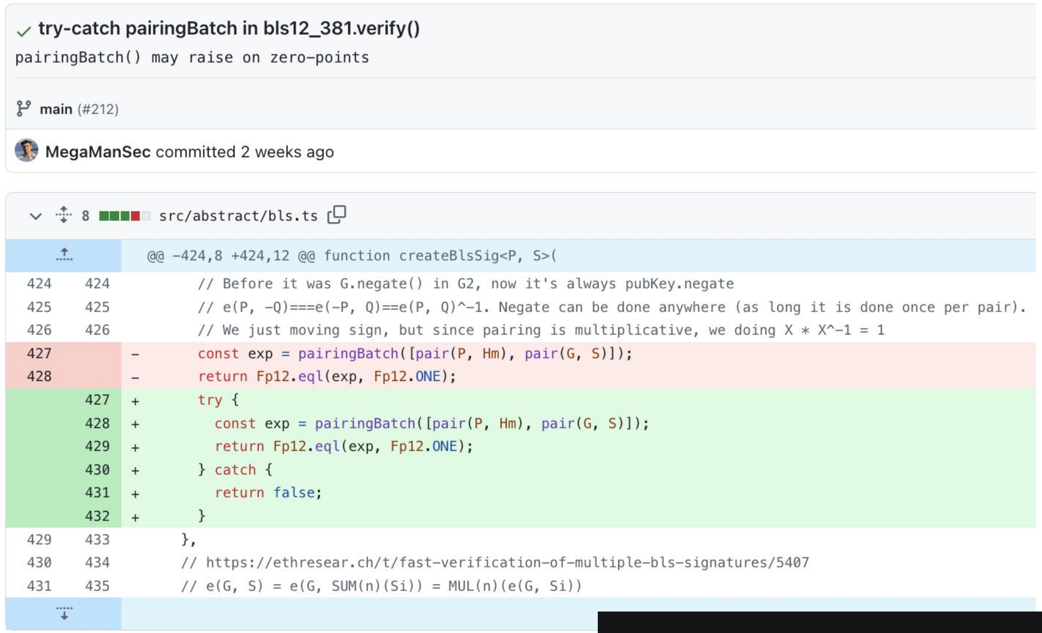
logfile_mod_udp_open() mistreated successful comm_connect_addr() result as an "Unable to connect" failure (and vice versa), rendering UDP-based logging unusable. Broken since at least 2010 commit d938215.

Also fixed logfile_mod_udp_close() closing FD 0 after "Invalid UDP logging address" ERRORs during logfile_mod_udp_open().

“Logical” Findings

HACKDAY

noble-curves: reachable raise



✓ try-catch pairingBatch in bls12_381.verify()

```
pairingBatch() may raise on zero-points
```

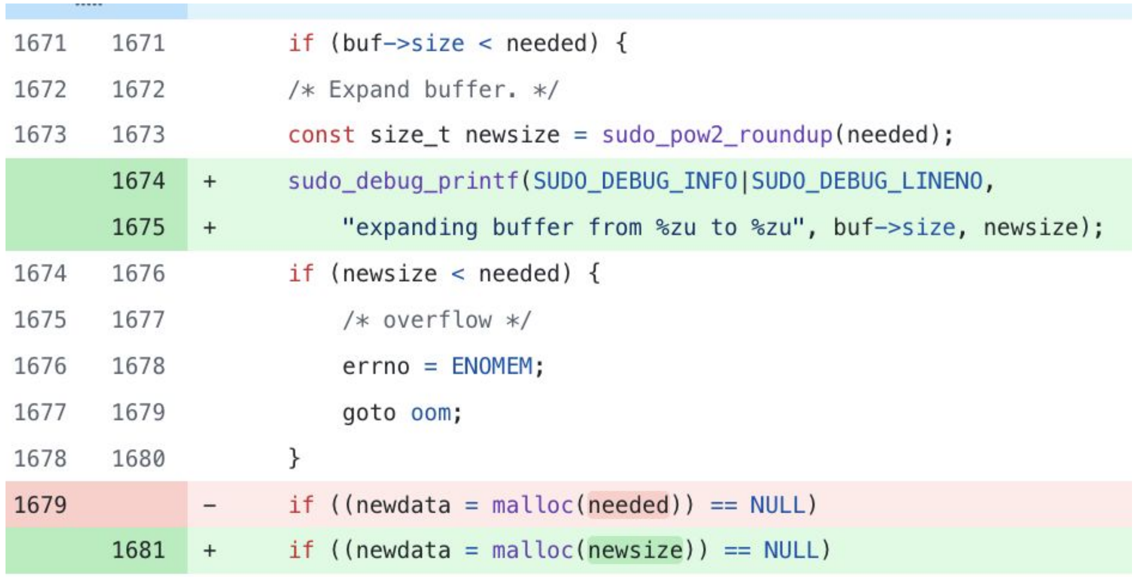
main (#212)

MegaManSec committed 2 weeks ago

8 src/abstract/bls.ts

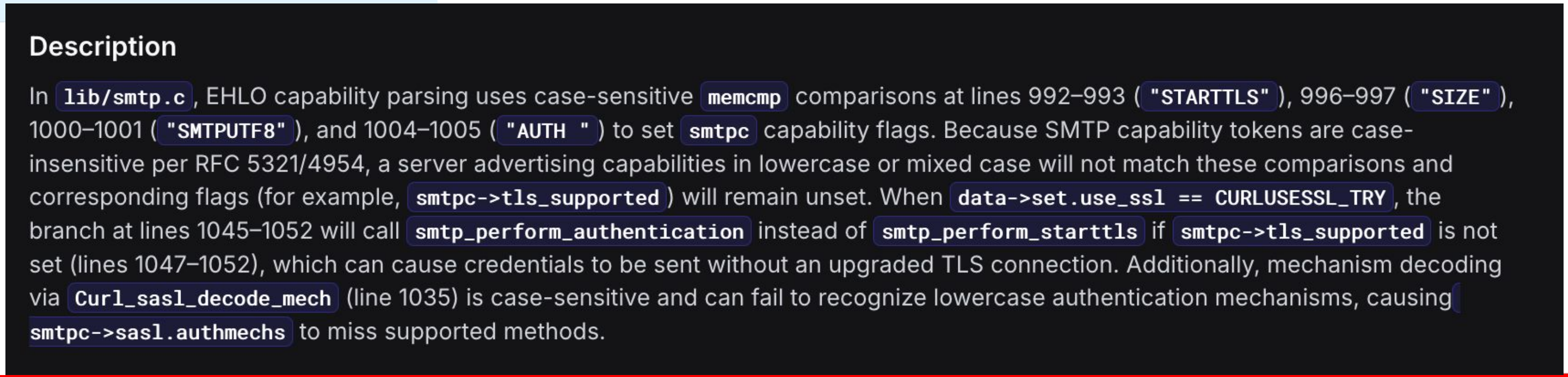
```
@@ -424,8 +424,12 @@ function createBlsSig<P, S>(
         // Before it was G.negate() in G2, now it's always pubKey.negate
         // e(P, -Q)===e(-P, Q)==e(P, Q)^-1. Negate can be done anywhere (as long it is done once per pair).
         // We just moving sign, but since pairing is multiplicative, we doing X * X^-1 = 1
-        const exp = pairingBatch([pair(P, Hm), pair(G, S)]);
-        return Fp12.eql(exp, Fp12.ONE);
+        try {
+          const exp = pairingBatch([pair(P, Hm), pair(G, S)]);
+          return Fp12.eql(exp, Fp12.ONE);
+        } catch {
+          return false;
+        }
       },
       // https://ethresear.ch/t/fast-verification-of-multiple-bls-signatures/5407
       // e(G, S) = e(G, SUM(n)(Si)) = MUL(n)(e(G, Si))
```

sudo: incorrect buffer allocation



```
      if (buf->size < needed) {
      /* Expand buffer. */
      const size_t newsize = sudo_pow2_roundup(needed);
+     sudo_debug_printf(SUDO_DEBUG_INFO|SUDO_DEBUG_LINENO,
+         "expanding buffer from %zu to %zu", buf->size, newsize);
      if (newsize < needed) {
          /* overflow */
          errno = ENOMEM;
          goto oom;
      }
-     if ((newdata = malloc(needed)) == NULL)
+     if ((newdata = malloc(newsize)) == NULL)
```

curl: SMTP keywords bug



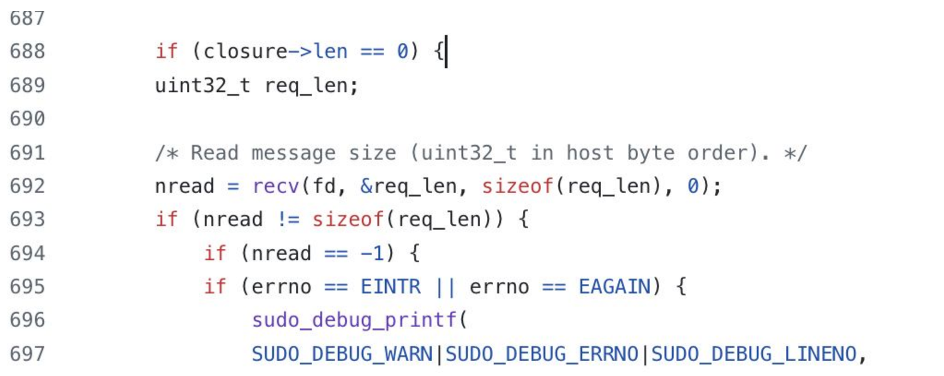
Description

In `lib/smtp.c`, EHLO capability parsing uses case-sensitive `memcmp` comparisons at lines 992–993 (`"STARTTLS"`), 996–997 (`"SIZE"`), 1000–1001 (`"SMTPUTF8"`), and 1004–1005 (`"AUTH "`) to set `smtpc` capability flags. Because SMTP capability tokens are case-insensitive per RFC 5321/4954, a server advertising capabilities in lowercase or mixed case will not match these comparisons and corresponding flags (for example, `smtpc->tls_supported`) will remain unset. When `data->set.use_ssl == CURLUSESSL_TRY`, the branch at lines 1045–1052 will call `smtp_perform_authentication` instead of `smtp_perform_starttls` if `smtpc->tls_supported` is not set (lines 1047–1052), which can cause credentials to be sent without an upgraded TLS connection. Additionally, mechanism decoding via `Curl_sasl_decode_mech` (line 1035) is case-sensitive and can fail to recognize lowercase authentication mechanisms, causing `smtpc->sasl.authmechs` to miss supported methods.

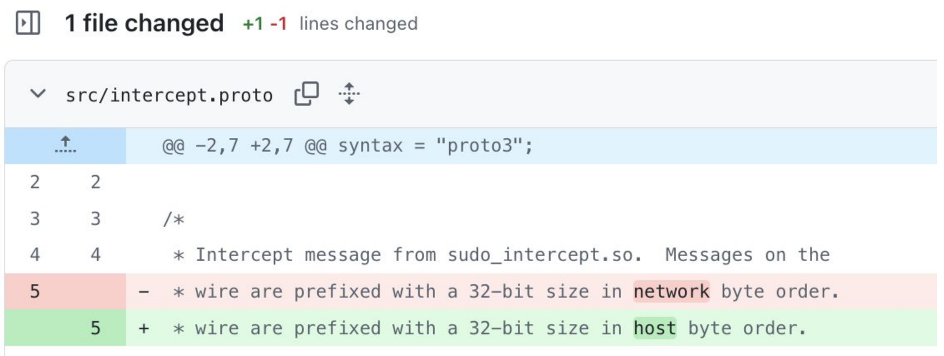
More “Logical” Findings

HACKDAY

sudo: host⟷network byte order?



```c
    if (closure->len == 0) {
    uint32_t req_len;

    /* Read message size (uint32_t in host byte order). */
    nread = recv(fd, &req_len, sizeof(req_len), 0);
    if (nread != sizeof(req_len)) {
        if (nread == -1) {
        if (errno == EINTR || errno == EAGAIN) {
            sudo_debug_printf(
            SUDO_DEBUG_WARN|SUDO_DEBUG_ERRNO|SUDO_DEBUG_LINENO,
```

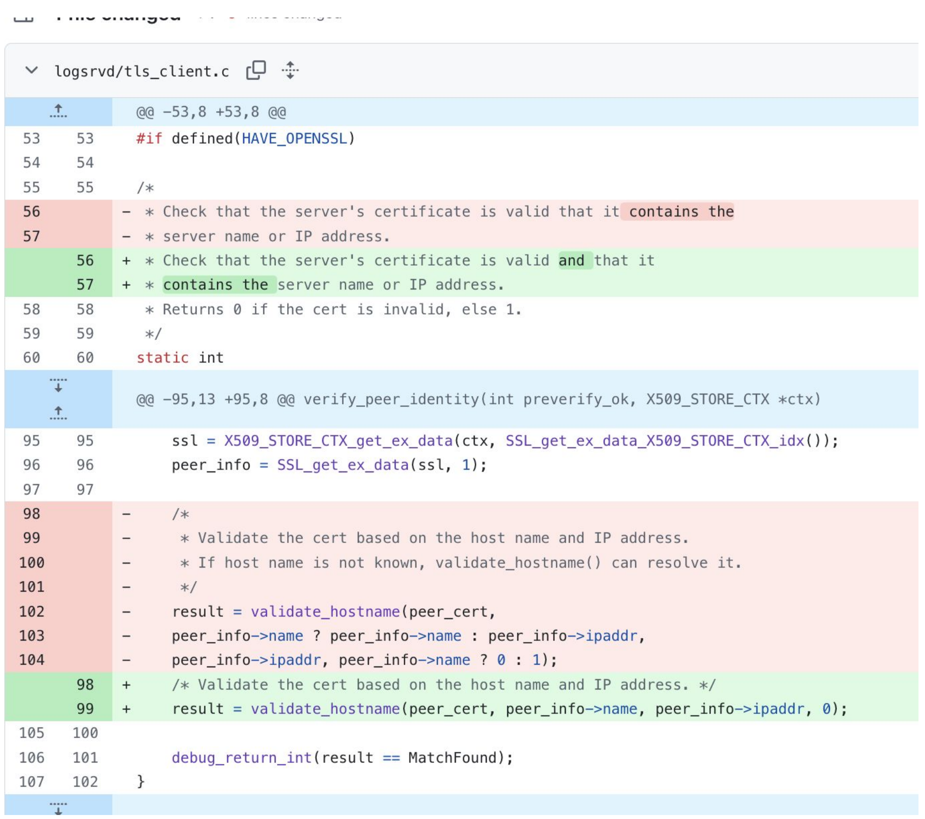


1 file changed +1 -1 lines changed

src/intercept.proto

```diff
@@ -2,7 +2,7 @@ syntax = "proto3";

 /*
  * Intercept message from sudo_intercept.so.  Messages on the
- * wire are prefixed with a 32-bit size in network byte order.
+ * wire are prefixed with a 32-bit size in host byte order.
```

sudo: TLS cert check for remote logging



logsrvd/tls_client.c

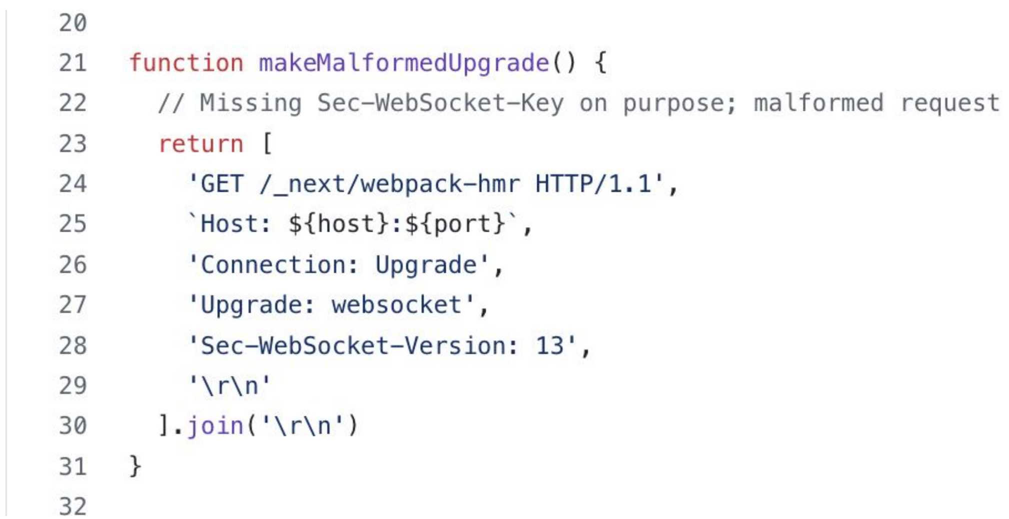
```diff
@@ -53,8 +53,8 @@
 #if defined(HAVE_OPENSSL)

 /*
- * Check that the server's certificate is valid that it contains the
- * server name or IP address.
+ * Check that the server's certificate is valid and that it
+ * contains the server name or IP address.
  * Returns 0 if the cert is invalid, else 1.
  */
 static int
@@ -95,13 +95,8 @@ verify_peer_identity(int preverify_ok, X509_STORE_CTX *ctx)
     ssl = X509_STORE_CTX_get_ex_data(ctx, SSL_get_ex_data_X509_STORE_CTX_idx());
     peer_info = SSL_get_ex_data(ssl, 1);

-    /*
-     * Validate the cert based on the host name and IP address.
-     * If host name is not known, validate_hostname() can resolve it.
-     */
-    result = validate_hostname(peer_cert,
-    peer_info->name ? peer_info->name : peer_info->ipaddr,
-    peer_info->ipaddr, peer_info->name ? 0 : 1);
+    /* Validate the cert based on the host name and IP address. */
+    result = validate_hostname(peer_cert, peer_info->name, peer_info->ipaddr, 0);

     debug_return_int(result == MatchFound);
 }
```

Logging to https://1.1.1.1/ or https://logging.com/. Connect.
if https certificate is for example.com, check if example.com resolves to 1.1.1.1 or the same IP address as logging.com.
if yes, certificate pass.

Other Languages

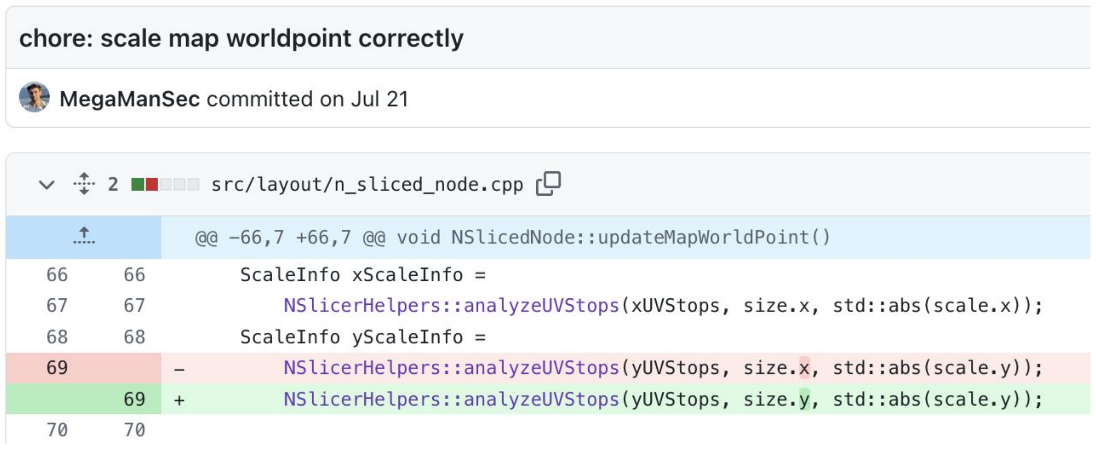
HACKDAY

next.js: memory/socket leak on websocket upgrade



```
function makeMalformedUpgrade() {
  // Missing Sec-WebSocket-Key on purpose; malformed request
  return [
    'GET /_next/webpack-hmr HTTP/1.1',
    `Host: ${host}:${port}`,
    'Connection: Upgrade',
    'Upgrade: websocket',
    'Sec-WebSocket-Version: 13',
    '\r\n'
  ].join('\r\n')
}
```

Incorrect map building in c++ (x,y mixup)



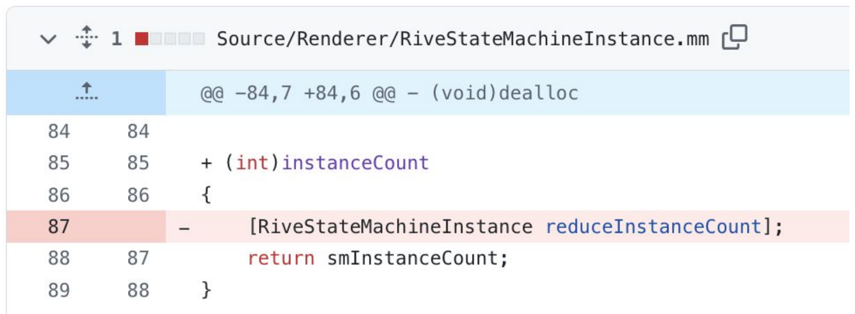
chore: scale map worldpoint correctly

MegaManSec committed on Jul 21

2 src/layout/n_sliced_node.cpp

```
@@ -66,7 +66,7 @@ void NSlicedNode::updateMapWorldPoint()
     ScaleInfo xScaleInfo =
         NSlicerHelpers::analyzeUVStops(xUVStops, size.x, std::abs(scale.x));
     ScaleInfo yScaleInfo =
-        NSlicerHelpers::analyzeUVStops(yUVStops, size.x, std::abs(scale.y));
+        NSlicerHelpers::analyzeUVStops(yUVStops, size.y, std::abs(scale.y));
```

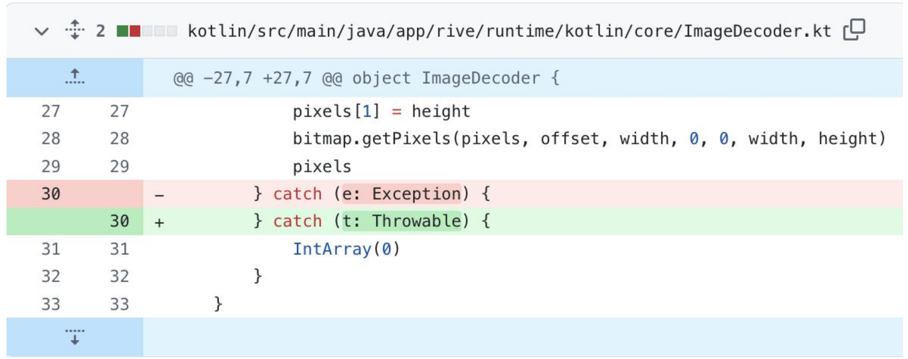
Incorrect refcount code in Obj-C



1 Source/Renderer/RiveStateMachineInstance.mm

```
@@ -84,7 +84,6 @@ - (void)dealloc

 + (int)instanceCount
 {
-    [RiveStateMachineInstance reduceInstanceCount];
     return smInstanceCount;
 }
```

Incorrect Except/Throw catch in Kotlin



2 kotlin/src/main/java/app/rive/runtime/kotlin/core/ImageDecoder.kt

```
@@ -27,7 +27,7 @@ object ImageDecoder {
                pixels[1] = height
                bitmap.getPixels(pixels, offset, width, 0, 0, width, height)
                pixels
-            } catch (e: Exception) {
+            } catch (t: Throwable) {
                IntArray(0)
            }
        }
```

HACKDAY

Operationalizing

How to effectively use these tools.

Patterns That Win (For Pentesters)

1/ Treat these systems like human code reviewers.

2/ Provide meaningful input.

3/ Guide them with policies.

4/ Pipe difficult-to-understand issues into ChatGPT.

Patterns That Win (For Sec. Teams)

1/ Treat these systems like human code reviewers.
2/ Provide meaningful input.
3/ Guide them with policies.
4/ Periodic full scans; embrace non-determinism.
5/ PR scans; blocking on high-severity findings.
6/ Keep a human in the loop; treat auto-fixes as technical description.

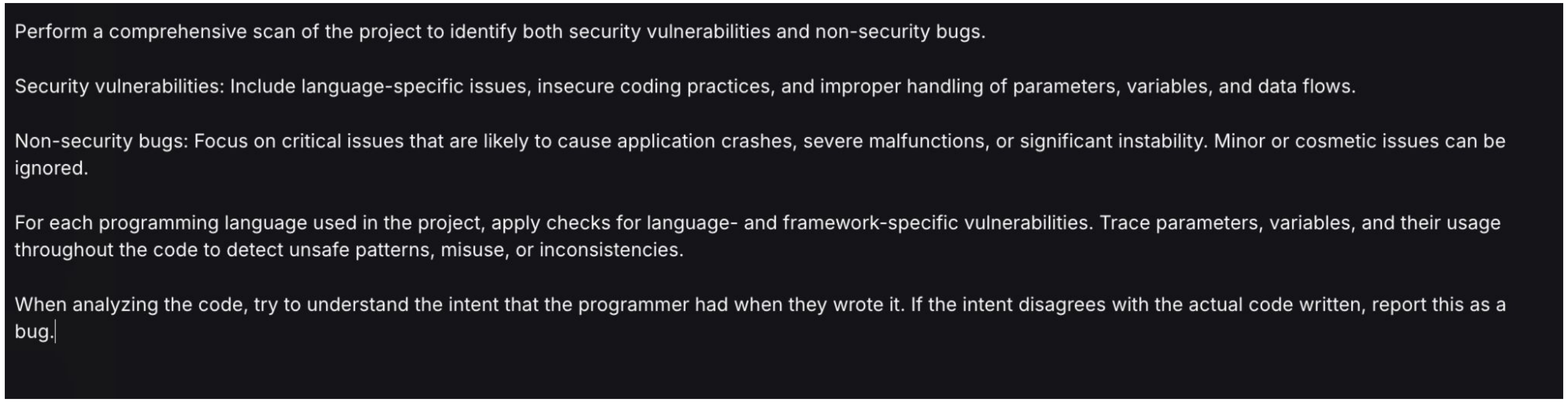
Testing Runbook

1/ Perform a full repo scan.

2/ Perform a full repo scan again.

3/ Perform a full repo scan with a rule/policy.

4/ Perform an (automatic) PR scan.



Perform a comprehensive scan of the project to identify both security vulnerabilities and non-security bugs.

Security vulnerabilities: Include language-specific issues, insecure coding practices, and improper handling of parameters, variables, and data flows.

Non-security bugs: Focus on critical issues that are likely to cause application crashes, severe malfunctions, or significant instability. Minor or cosmetic issues can be ignored.

For each programming language used in the project, apply checks for language- and framework-specific vulnerabilities. Trace parameters, variables, and their usage throughout the code to detect unsafe patterns, misuse, or inconsistencies.

When analyzing the code, try to understand the intent that the programmer had when they wrote it. If the intent disagrees with the actual code written, report this as a bug.

HACKDAY

Key Takeaways

So what?

Takeaways

- AI SASTs are real and extremely useful already.
- Especially strong at logic and architectural flaws.
- Made even stronger with policies.
- Biggest value IMO: finding inconsistencies between intent and implementation.
- Already beyond most SCAs.
- Pentesters: Use them for recon, easy findings, and odd-path exploration.
- Security/Product Teams: Integrate into PR checks, run periodically. Easy wins for old, large codebases.
- Try out Almanax, Corgea, ZeroPath (and DryRun, if possible).

KAZ HACK STAN 2025 SEPTEMBER 17–19 ALMATY

TSARKA · DIGITAL & SPACE MINISTRY · FREEDOM HOLDING CORP. · TREND MICRO · astana hub

HACKDAY

THANK YOU FOR YOUR ATTENTION

greetz

Joshua Rogers <mail: *hackday@joshua.hu*>