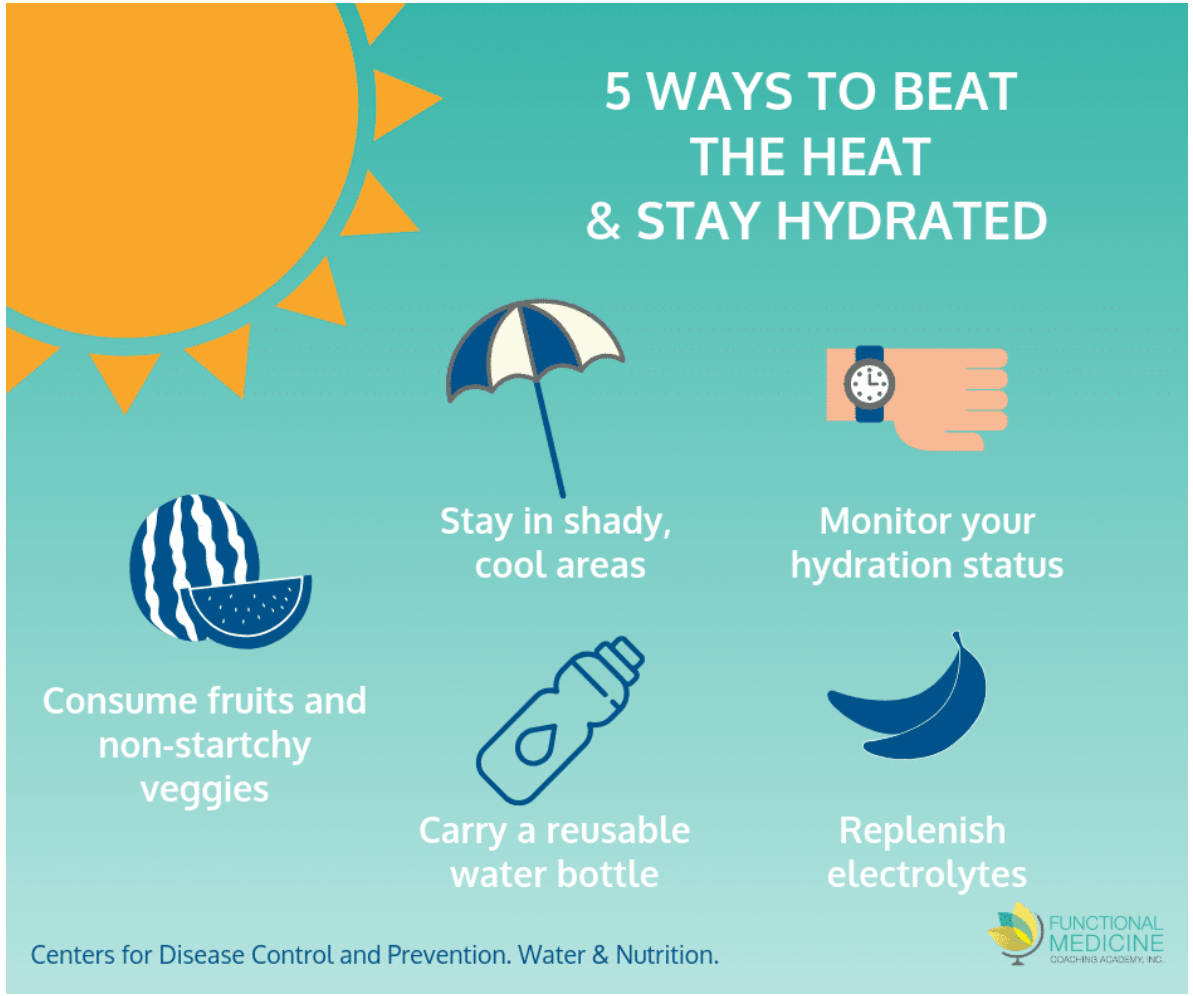
LUMI AI Factory

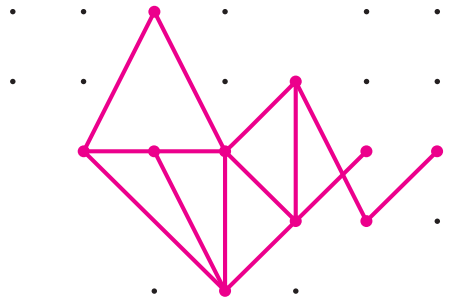
Securing GitHub & GitLab Repositories in the Era of Supply Chain Attacks and AI Agents

Let‘s keep us cool

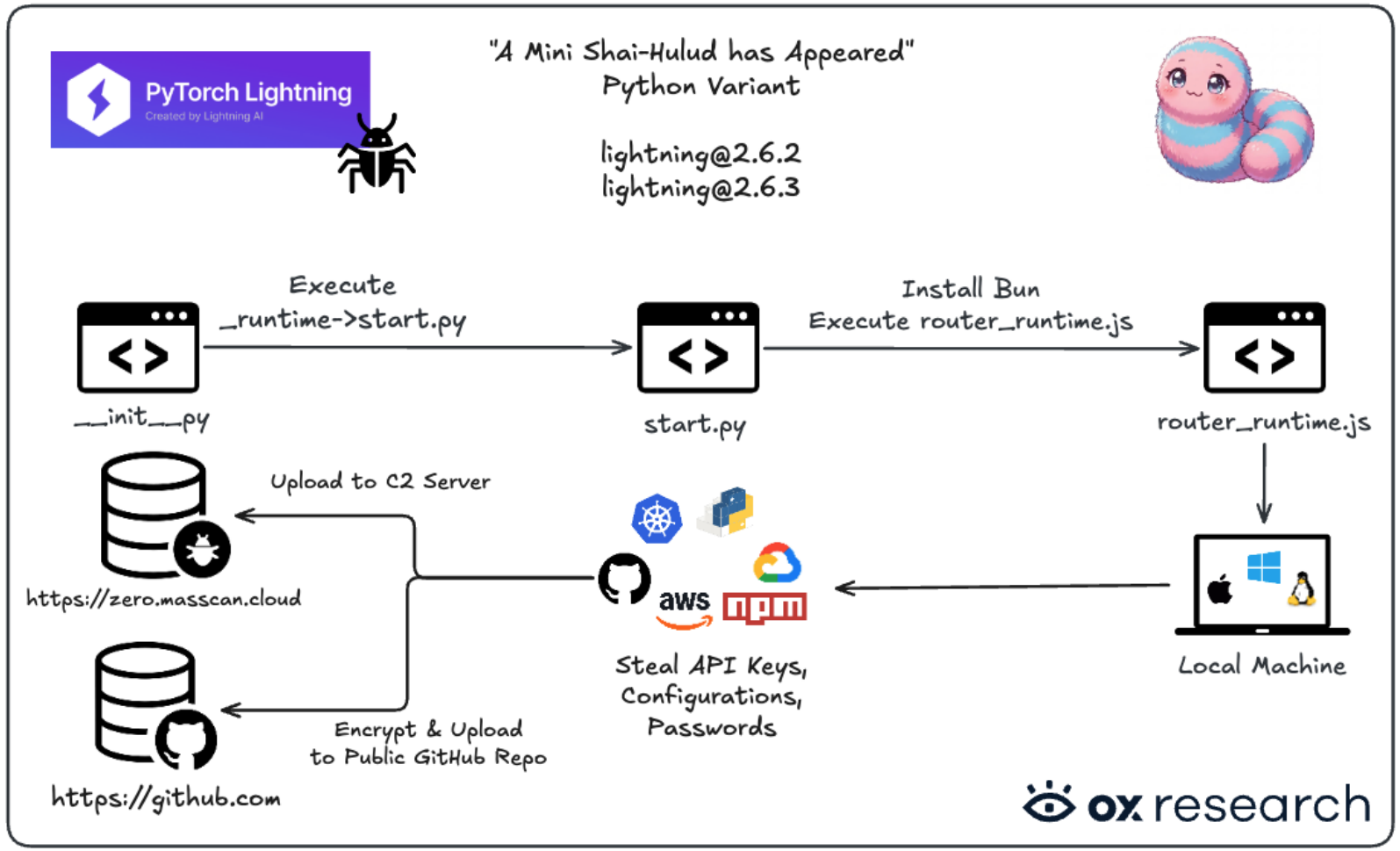


LUMI AI Factory

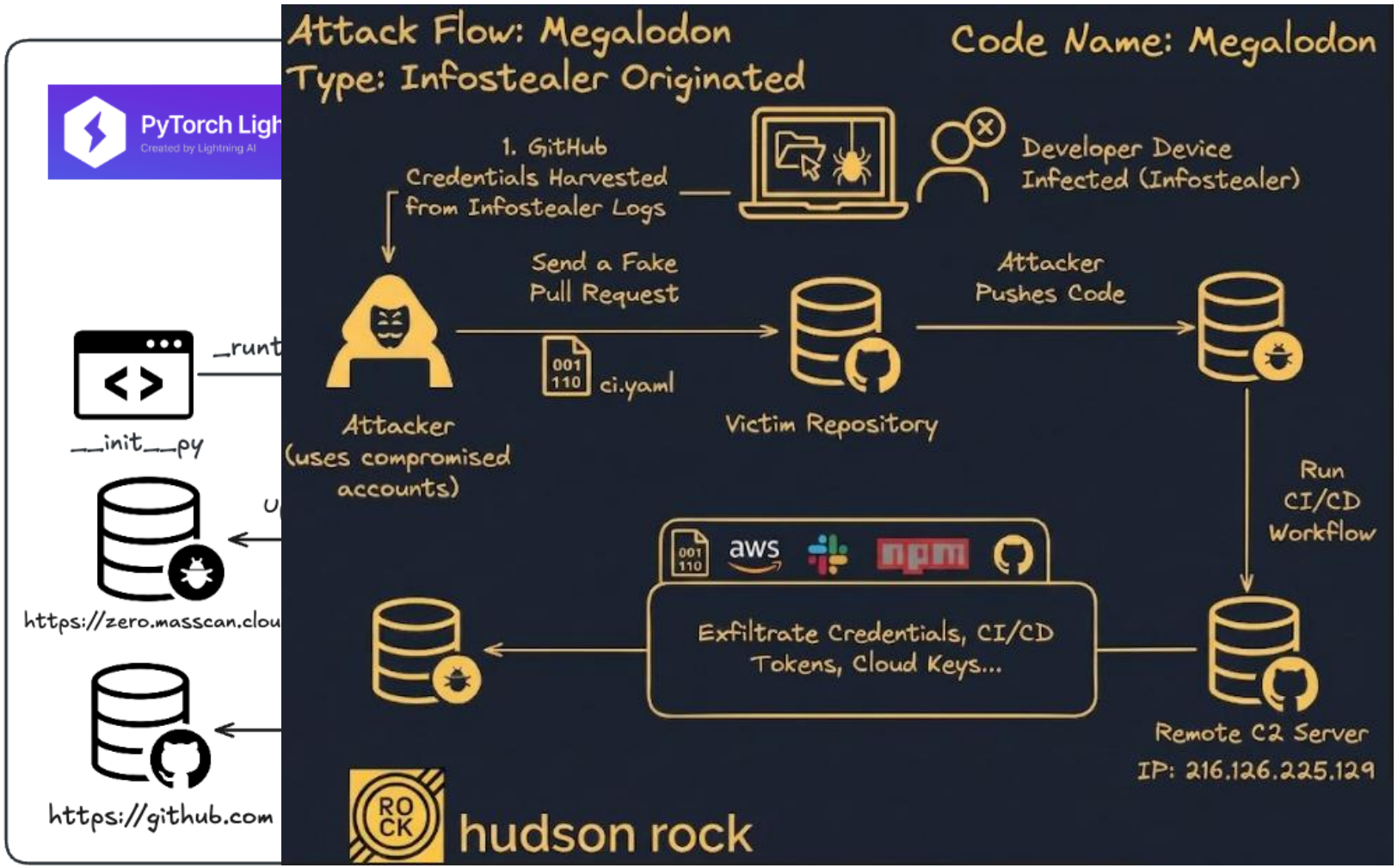
Motivation



Motivation

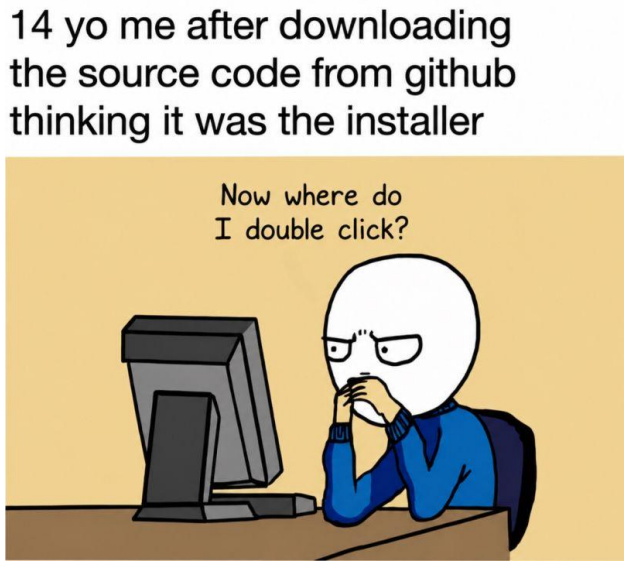

PyTorch Lightning
Created by Lightning AI
"A Mini Shai-Hulud has Appeared"
Python Variant
lightning@2.6.2
lightning@2.6.3
Execute
_runtime->start.py
Install Bun
Execute router_runtime.js
__init__.py
start.py
router_runtime.js
Upload to C2 Server
https://zero.masscan.cloud
aws
npm
Steal API Keys,
Configurations,
Passwords
Local Machine
Encrypt & Upload
to Public GitHub Repo
https://github.com
ox research

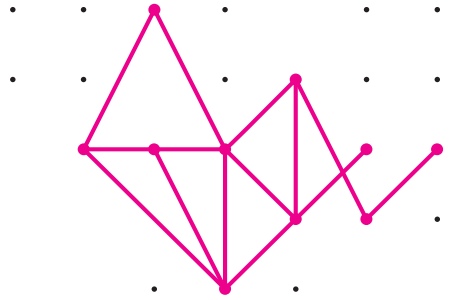
Motivation


PyTorch Ligh
Created by Lightning AI
_runt
__init__.py
https://zero.masscan.clou
https://github.com
Attack Flow: Megalodon
Type: Infostealer Originated
Code Name: Megalodon
1. GitHub
Credentials Harvested
from Infostealer Logs
Developer Device
Infected (Infostealer)
Send a Fake
Pull Request
ci.yaml
Attacker
Pushes Code
Attacker
(uses compromised
accounts)
Victim Repository
Run
CI/CD
Workflow
aws
npm
Exfiltrate Credentials, CI/CD
Tokens, Cloud Keys...
Remote C2 Server
IP: 216.126.225.129
hudson rock

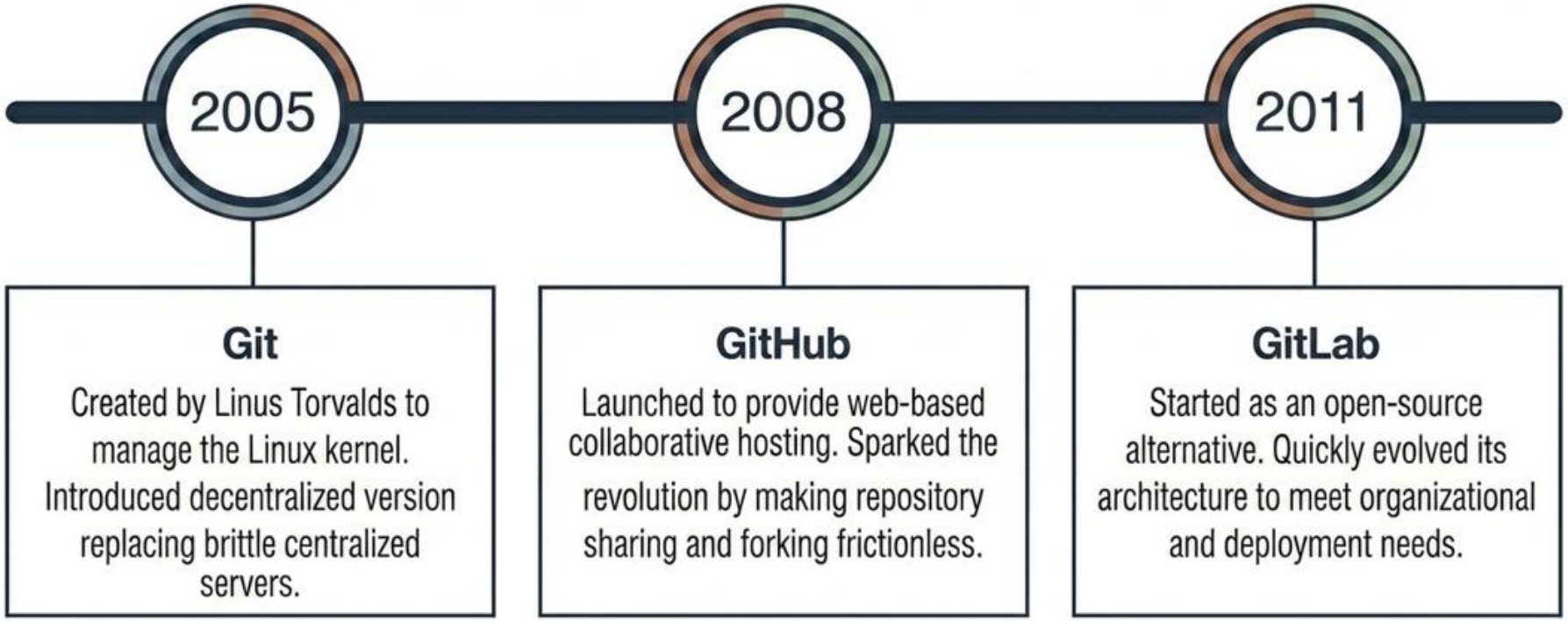
What can you expect

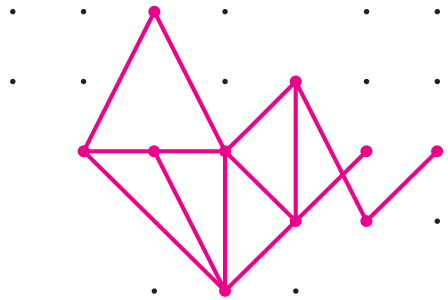
- Short history of Git
- Current threats and vulnerabilities
- AI agents in the workflow
- Best practices on how to secure repositories





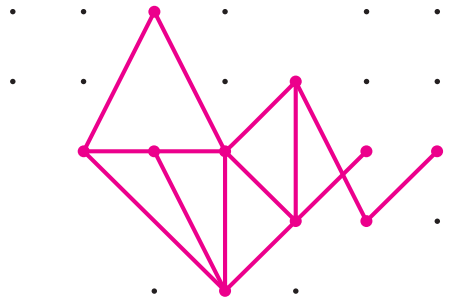
Git History


2005
2008
2011
Git
Created by Linus Torvalds to manage the Linux kernel. Introduced decentralized version replacing brittle centralized servers.
GitHub
Launched to provide web-based collaborative hosting. Sparked the revolution by making repository sharing and forking frictionless.
GitLab
Started as an open-source alternative. Quickly evolved its architecture to meet organizational and deployment needs.



GitHub vs GitLab

Aspect	GitHub	GitLab
Ownership & Model	Owned by Microsoft; SaaS-first with enterprise options	Independent; open-core with SaaS and self-hosted options
Core Strength	Collaboration, open-source community, integrations	End-to-end DevOps in a single platform
CI/CD	GitHub Actions, integration-driven	Native, deeply integrated CI/CD pipelines
Self-Hosting	Limited and enterprise-focused	Strong and widely adopted self-hosting support
Typical Use Cases	Open-source projects, startups, distributed teams	Enterprises, regulated environments, DevSecOps



TeamPCP

LUMI AI Factory

What the Malware Targeted

Credential categories targeted by the malicious binary embedded in trojanized packages.

CLOUD PROVIDER

AWS / Azure / GCP Credentials

Instance metadata endpoints, managed identity tokens, and service account keys from CI runners and developer workstations.

`Azure MSI endpoint` `GCP metadata API`

IDENTITY & ACCESS

GitHub Personal Access Tokens

Long-lived PATs grant write access to repositories — exactly how Trivy's compromise was used to access Checkmarx KICS.

`~/.gitconfig` `~/.git-credentials`

CI/CD TOKEN

GITHUB_TOKEN (Actions)

Ephemeral token injected into every Actions run. In many pipelines it has write access to packages, releases, and contents.

`env: GITHUB_TOKEN` `secrets.GITHUB_TOKEN`

PACKAGE REGISTRIES

npm, PyPI, Maven & Other Registry Tokens

Publish tokens let an attacker push trojanized versions of any package the developer maintains. A single stolen npm token can compromise every package owned by that account.

`~/.npmrc` `~/.pypirc` `env: NPM_TOKEN` `env: PYPI_API_TOKEN`

PRIVATE KEYS

SSH Private Keys

Unencrypted SSH keys provide persistent access to production infrastructure and internal version control systems.

`~/.ssh/id_rsa` `~/.ssh/id_ed25519`

AI / LLM APIS

LLM Service API Keys

Keys for OpenAI, Anthropic, and similar services found in developer dotfiles and CI runners. Useful for uncapped compute abuse.

`~/.claude.json`

`~/.kiro/settings/mcp.json`

CONTAINER & ORCHESTRATION

Container & Kubernetes Tokens

Docker Hub credentials and Kubernetes service account JWTs enable lateral movement into production container infrastructure.

`~/.docker/config.json`

`/var/run/secrets/kubernetes.io`

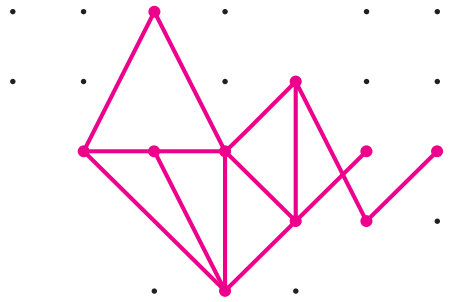
BROAD SWEEP

Environment Variable Pattern Matching

Broad sweep of all environment variables matching common secret naming patterns — catching any service not specifically targeted above.

`*_TOKEN` `*_SECRET` `*_KEY`

Okta report



TeamPCP

LUMI AI Factory

S Security Analyst

⚠️ CRITICAL SECURITY ALERT: Shai-Hulud v2 Malware Detected

YES - Your project is severely affected by the Shai-Hulud v2 malware campaign.

🚨 Immediate Threat Assessment

I've detected **689 HIGH severity vulnerabilities** in your `package-lock.json` file, ALL related to the Shai-Hulud v2 npm supply chain attack. This is an active, widespread malware campaign targeting npm packages through account takeovers.

Affected Packages (Sample)

Your project contains numerous compromised packages including:

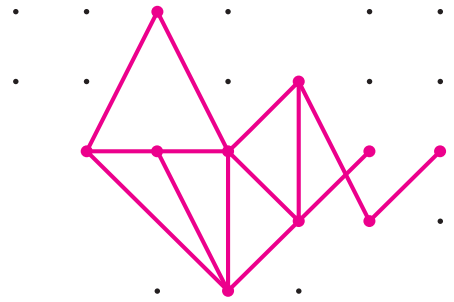
- **@ensdomains/** packages (ens-contracts, ens-avatar, ensjs, thorin, etc.)
- **@voiceflow/** packages (logger, common, metrics, etc.)
- **@posthog/** packages (clickhouse, cli, ai, icons, etc.)
- **@asyncapi/** packages (keeper, web-component, studio, etc.)
- **@oku-ui/** packages (slider, accordion, primitives, etc.)
- **@actbase/** packages (react-kakaosdk, react-native-* packages)
- **@browserbasehq/** packages (mcp-server-browserbase, stagehand, etc.)
- And **hundreds more** across various npm organizations

Attack Vector

This campaign involves:

- **NPM account takeovers** of legitimate package maintainers
- **Malicious code injection** into trusted packages
- **Credential theft** and **data exfiltration** capabilities
- **Supply chain compromise** affecting downstream users

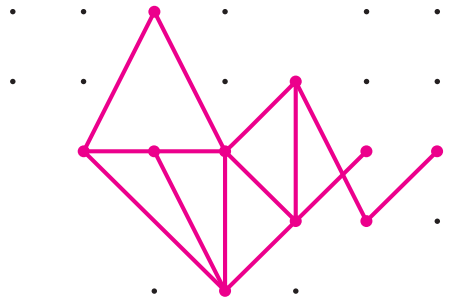
GitLab Security Analyst



Pull request target from the hell

- pull_request_target runs with base repository privileges ⚠️
- It exposes secrets and write permissions
- Never execute untrusted PR code
- A single forked PR can lead to RCE
- Use only when necessary

```yaml
name: Labeler
on: [pull_request_target]

jobs:
  label:

    runs-on: ubuntu-latest
    permissions:
      contents: read
      pull-requests: write

    steps:
    - uses: actions/labeler@v4
      with:
        repo-token: "${{ secrets.GITHUB_TOKEN }}"
```

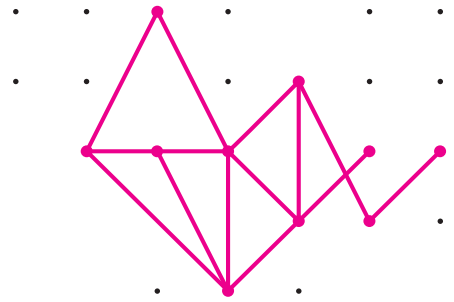


LUMI AI Factory

Pull request target from the hell

```yaml
# Run only when the PR comes from the same repository, not from a fork
if: github.event.pull_request.head.repo.full_name == github.repository

# Run when a label is added to a PR
on:
  pull_request_target:
    types: [labeled]

---

# Run only when the PR is labeled "verified"
if: contains(github.event.pull_request.labels.*.name, 'verified')

# protections, secrets and required reviewers are applied
environment: testing
```

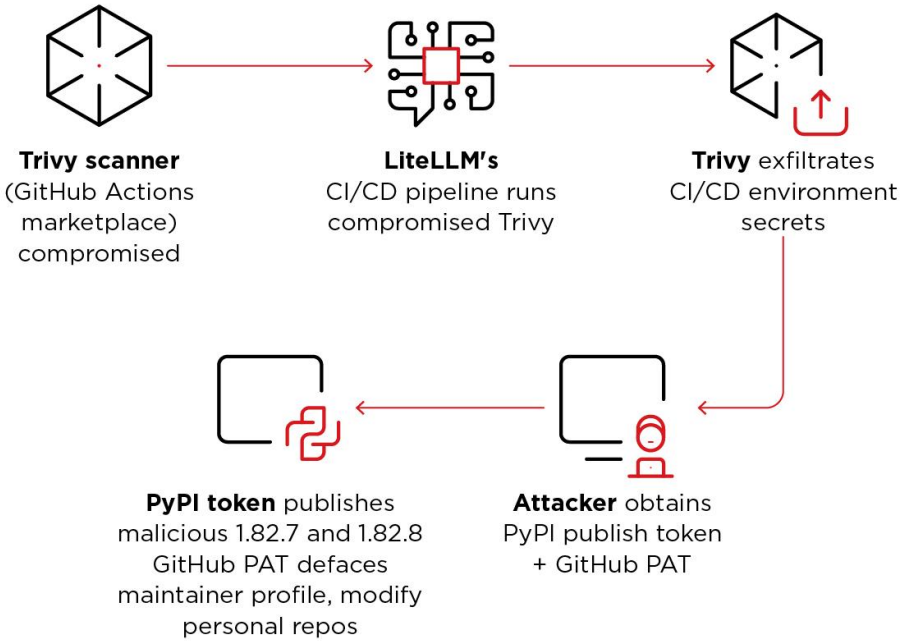
Blog by Roi Nisimi



LUMI AI Factory

LiteLLM horror story

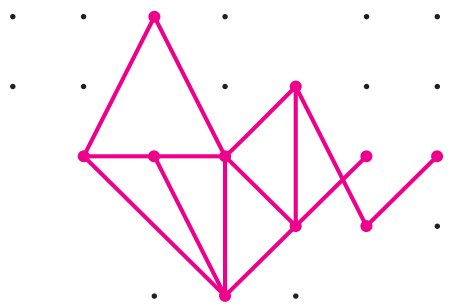
- Popular AI proxy package (100+ LLMs)
- Attack is linked to TeamPCP (Shai-Hulud)
- Started within the trivy-action repo
- GitHub Actions were compromised
- The malicious code stole the SSH keys
- Two versions were published to PyPi



Trend Micro blog

LiteLLM horror story

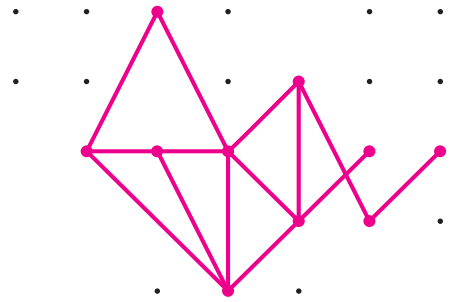
- The malicious versions deployed a three-stage payload:
  - Credential harvester
  - Kubernetes lateral movement toolkit
  - Persistent backdoor
- Callum McMahon at FutureSearch discovered the payload
- The highest-priority IOCs can be found in this link.



LUMI AI Factory

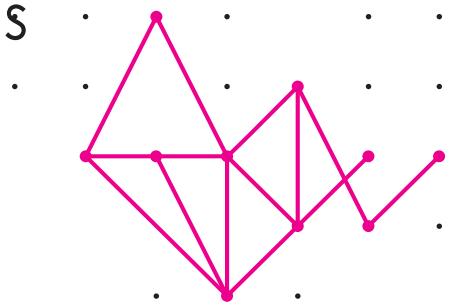
LiteLLM horror story

- Check for infected files and directories
- Rotate ALL credentials
- Block the known IOCs at DNS and firewall
- Audit npm packages
- Pin Trivy and all security scanners in your CI/CD pipeline



LiteLLM horror story

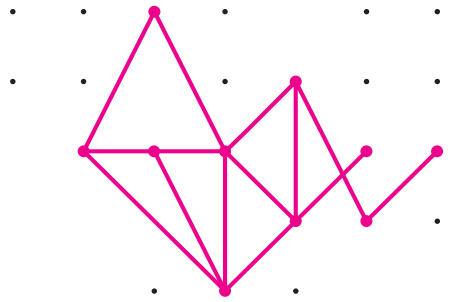
- Enforce ‘npm install --ignore-scripts’ in CI/CD pipelines
- Monitor for unexpected ‘.pth’ file creation in Python site-packages
- Pin package versions and verify SHA256 hashes
- Pin to a stable and tried-and-tested version
- Deploy behavioral detection for bulk credential file reads
- Implement JARM fingerprint monitoring for TLS connections



LUMI AI Factory

LiteLLM horror story

- Audit transitive dependencies
- Implement egress filtering
- Treat LLM API keys as crown jewels
- Watch for fork bomb / resource exhaustion

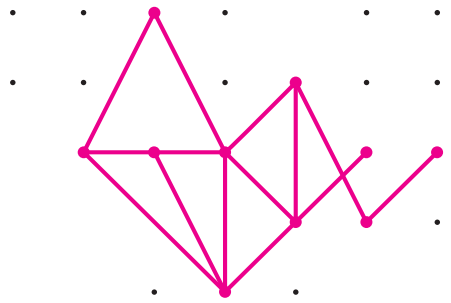


AI Agents in CI/CD pipelines

LUMI AI Factory

- In both GitHub Actions and GitLab CI/CD pipelines
- AI agents like Gemini CLI, Claude Code, OpenAI Codex, …
- Untrusted user content can be embedded directly into prompts
- AI output is executed as shell commands

```
prompt: |
  Analyze this issue:
  Title: "${{ github.event.issue.title }}"
  Body: "${{ github.event.issue.body }}"
```



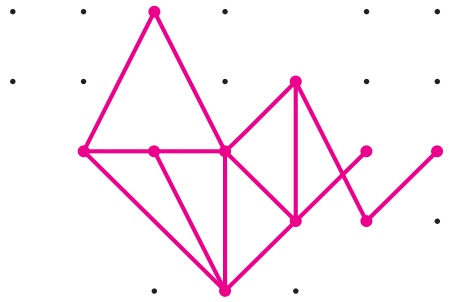
AI Agents in CI/CD pipelines

Aikido Blog post

AI Agents in CI/CD pipelines

LUMI AI Factory

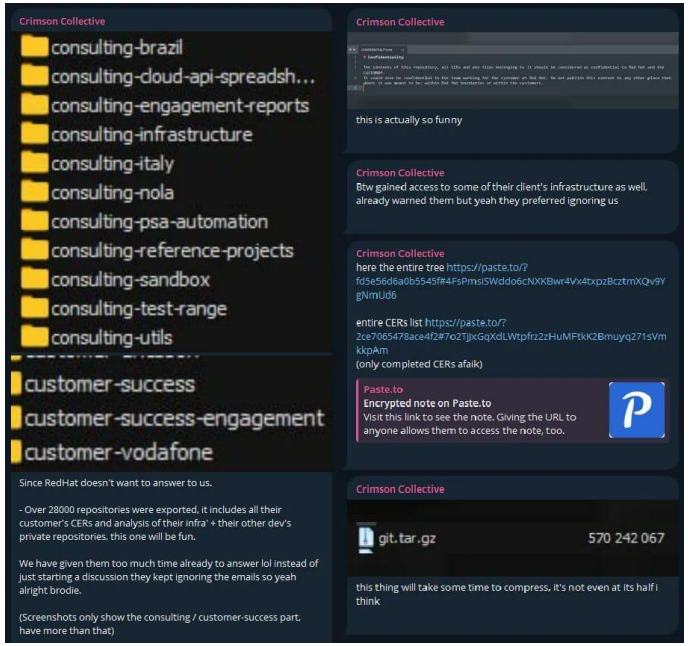
- Restrict the toolset available to AI agents
- Avoid injecting untrusted user input into AI prompts
- Treat AI output as untrusted code
- Restrict the blast radius of leaked GitHub tokens



LUMI AI Factory

Red Hat GitLab breach

- Nearly 570GB of compressed data was stolen
- Almost 800 organizations were affected
- The Crimson Collective - cyber extortion group

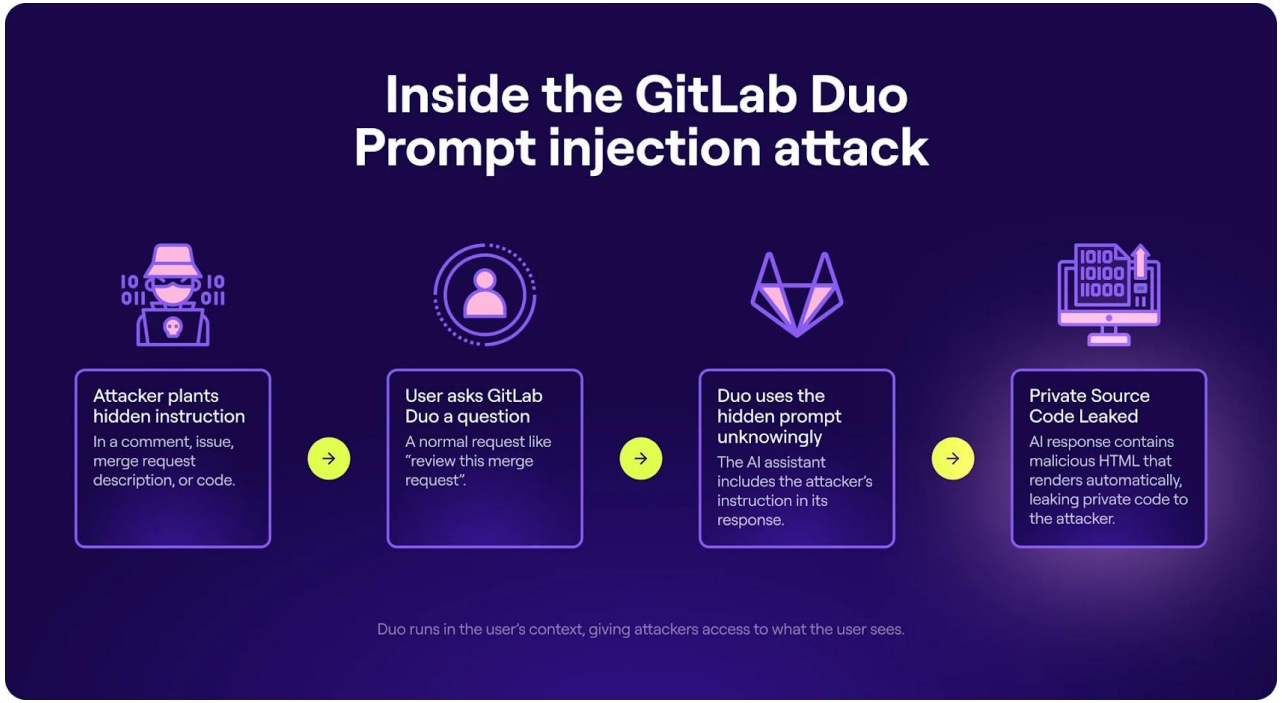


The GitGuardian

GitLab Duo Vulnerability

LUMI AI Factory

- GitLab Duo is an AI-powered coding assistant
- Duo Chat has been susceptible to an indirect prompt injection

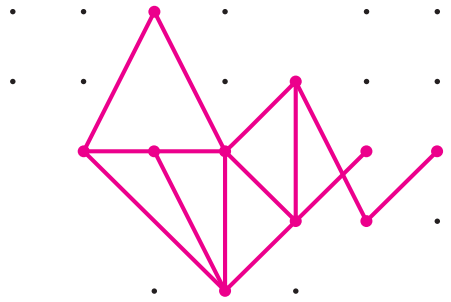


The HackerNews

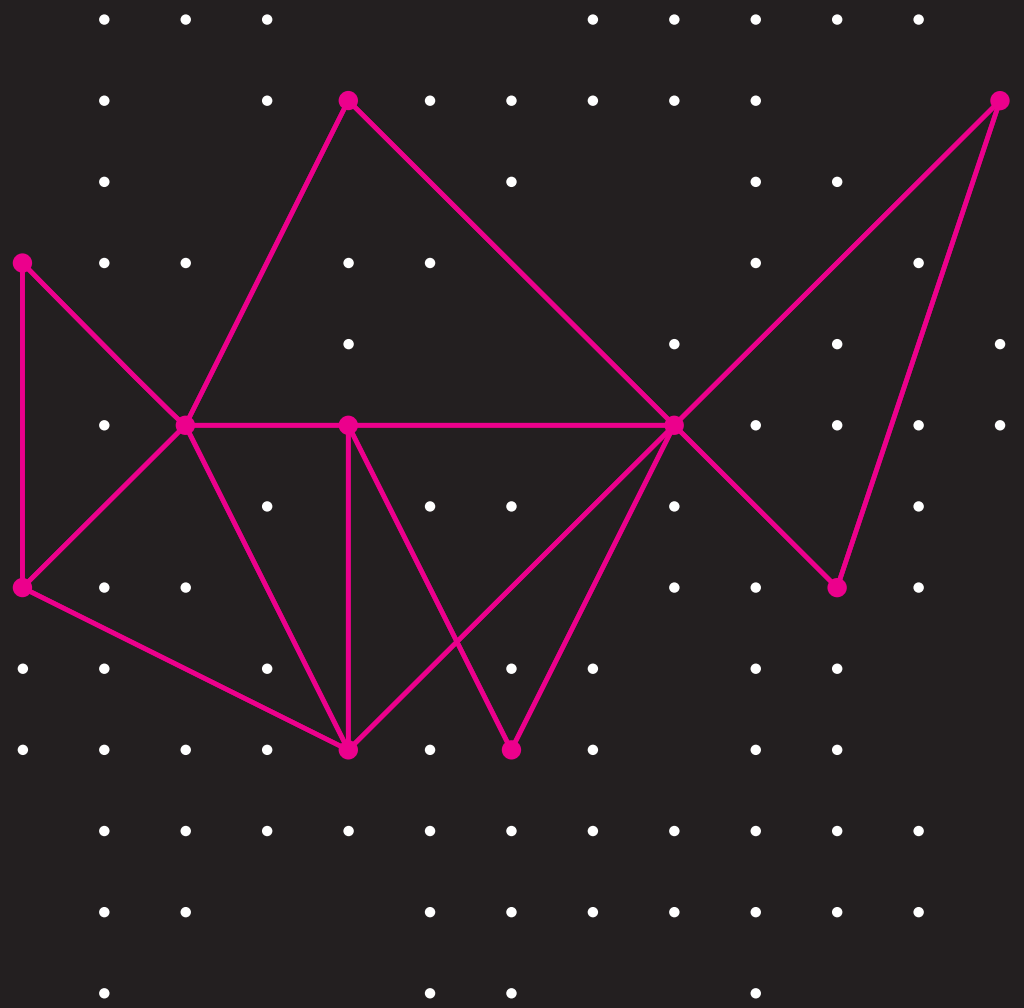
LUMI AI Factory

Conclusion

- Protect CI/CD as carefully as production
- AI agents are both an opportunity and a new attack surface
- Security must be built into the development workflow



LUMI AI Factory



LUMI AI Factory

Ing. Dominika Regéciová

Dominika.Regeciova@vsb.cz

www.vsb.cz