GMO Flatt Security | Takumi by GMO

絶え間ない脆弱性ニュースの裏で止まらない悪性パッケージ公開



1,900万検体の解析から見る ソフトウェアサプライチェーン攻撃の実態

2026/11/17

CODE BLUE 2026

GMO Flatt Security 株式会社 執行役員 豊田 恵二郎



豊田 恵二郎 @toyojuni

- GMO Flatt Security（株） 執行役員 CCO / 共同創業者
- 東京大学工学部航空宇宙工学科卒
- 事業責任者として P/L に責任を持ちつつ、
  広報・マーケティングを管掌

2026年10月21日（水）、PHP ニュースがお茶の間をも席巻する裏で



日中

php2shell の情報が出回る。攻撃コードとされるものも流通

11:12

当社ブログで速報を公開

22:40 頃

PHP の修正版リリース

8.2.35 / 8.3.36

8.4.27 / 8.5.12



19:44

litellm==1.106.2
正規版を公開

20:39

litellm==1.107.0
悪性版を公開

21:27

当社ブログで速報を公開

同日中

PyPI が削除

LiteLLMは月間約1億DL

時刻はいずれも 2026-10-21 の日本時間。出典: 当社ブログ（php2shell、LiteLLM 再侵害）

AIを駆使して、脆弱性と悪性への対処をお支えします

AI脆弱性診断・ペネトレーションテスト



脆弱性報告数
16万件突破

ソフトウェアサプライチェーン対策



日々の検査数
6,000万件突破

脆弱性関連ニュースが相次ぐ中でも、悪性ニュースも健在。
そして、ニュースになるものは氷山の一角でしかない



報道される個別事案

日々公開される
大量のパッケージ

半年間に公開されたパッケージを全量解析して確かめた

解析期間：2026年2月19日〜8月18日（181日間、6レジストリ）

19,213,180検体

観測期間中に解析したパッケージ名＋バージョン

31,840検体（0.2%）

期間内に公開され、悪性と判定したもの

01 **報道された侵害** 入口は公開者と CI に移った

02 **報道のない日** 悪性パッケージは毎日公開されていた

03 **届き方と狙い** 型と活動群から読み解く

04 **10月の3事案** 攻撃は部品化して続いている

05 **自社を守るヒント** 止められない前提で層を重ねる

PART 1

報道された侵害

入口は公開者と CI に移った

2026年、正規の公開経路から悪性版が届く事案が毎月のように続いた

2月

2月〜3月
Trivy
Action のタグと公式リリースを改変

3月

3/24・3/27
LiteLLM・Telnyx
Trivy で漏れたクレデンシャルが起点とみられる

3/31
axios
週1億 DL 超。依存1行の追加で RAT

4月

4/23
@bitwarden/cli
公開用ワークフローを改変

4/29〜5/11
Mini Shai-Hulud・TanStack
ワームが npm と PyPI に拡散

5月

5月
Laravel-Lang
Composer の autoload.files で発火

6月

6/17
Mastra
依存注入

7月

7/14
AsyncAPI
import 時の発火

8月

8/4
keyv
週1億 DL 超。ワームで多数を再公開

2026年、正規の公開経路から悪性版が届く事案が毎月のように続いた

2月

2月〜3月
Trivy
Action のタグと公式リリースを改変

3月

3/24・3/27
LiteLLM・Telnyx
Trivy で漏れたクレデンシャルが起点とみられる

3/31
axios
週1億 DL 超。依存1行の追加で RAT

4月

4/23
@bitwarden/cli
公開用ワークフローを改変

4/29〜5/11
Mini Shai-Hulud・TanStack
ワームが npm と PyPI に拡散

5月

5月
Laravel-Lang
Composer の autoload.files で発火

6月

6/17
Mastra
依存注入

7月

7/14
AsyncAPI
import 時の発火

8月

8/4
keyv
週1億 DL 超。ワームで多数を再公開

知られていないかもですが
広く使われており推移的依存が
わかりやすい

```
$ npm explain keyv
```

エコシステム側も、入り込む余地を減らす対策を急いでいる

npm の例

2023年9月	2025年7月	2025年12月	2026年2月	2026年5月	2026年7月
npm Provenance	**npm Trusted Publishing**	**npm 長期トークン廃止**	**cooldown**	**Staged Publishing**	**install script の既定停止**
どの CI がビルドしたかを検証	長期トークンなしで CI から公開	なりすまし公開を減らす	公開直後の悪性版を踏まない	公開に人の承認を挟む	インストール時の実行を止める

Provenance と Trusted Publishing は「正規の CI が公開した」ことしか保証しない

- @bitwarden/cli@2026.4.0 は、改変した公開ワークフローから OIDC 経由で公開された
- TanStack では、未信頼 PR が汚染したキャッシュを正規のリリースジョブが復元した
- 記録が示すのは公開した CI であり、**その CI に入ったコードの正しさではない**

エコシステム側も、入り込む余地を減らす対策を急いでいる

npm の例

2023年9月
npm Provenance
どの CI がビルドしたかを検証

2025年7月
npm Trusted Publishing
長期トークンなしで CI から公開

2025年12月
npm 長期トークン廃止
なりすまし公開を減らす

2026年2月
cooldown
公開直後の悪性版を踏まない

2026年5月
Staged Publishing
公開に人の承認を挟む

2026年7月
install script の既定停止
インストール時の実行を止める

npm v12（2026/07〜）では依存の install script が既定で止まるが、右側の経路は対象外である

※下記の例は npm に限らない

インストール	コマンド実行	ライブラリ読み込み	処理系の起動	リポジトリを開く
preinstall / postinstall axios、Mastra、keyv	**bin の差し替え** @bitwarden/cli	**import / require** AsyncAPI、Mini Shai-Hulud（PyPI）	**.pth / sitecustomize** LiteLLM（3月・10月） Hades（6月）	**開発ツールの設定** .claude/settings.json .vscode/tasks.json
binding.gyp Miasma（6月）	--ignore-scripts でも bw 実行で発火	**autoload.files** Laravel-Lang（Composer）	**import すら不要**	再実行と永続化に使われた

入口・改変・発火・通信のどの段階にも複数の経路がある

	入口		改変	
	正規公開者	Actions・CI	依存注入	本体・成果物
Trivy	・	●	・	●
axios	●	・	●	・
Bitwarden	・	●	・	●
Mini Shai-Hulud	●	・	・	●
TanStack	・	●	・	●
Mastra	●	・	●	・
AsyncAPI	・	●	・	●
keyv	●	・	・	●

●：主要な経路として確認したもの　・：主要な経路としては確認していないもの

出典: ソフトウェアサプライチェーン 脅威レポート 2026 図14

入口・改変・発火・通信のどの段階にも複数の経路がある

	発火		通信	
	導入時	import・require	専用 C2	GitHub 等の悪用
Trivy	●	・	●	・
axios	●	・	●	・
Bitwarden	●	・	●	●
Mini Shai-Hulud	●	●	●	●
TanStack	●	・	●	●
Mastra	●	・	●	・
AsyncAPI	・	●	●	●
keyv	●	・	●	●

●：主要な経路として確認したもの　・：主要な経路としては確認していないもの

出典: ソフトウェアサプライチェーン 脅威レポート 2026 図14

これら悪性キャンペーンは一球入魂な性質を帯びる

脆弱性を突く攻撃

攻撃者の手元で稼働するAIが状況に応じて柔軟な攻撃を繰り出せるので、場当たり的でも問題ない

悪性キャンペーン

まず定型スクリプトを被害者環境で安定稼働させる必要があり、(少なくとも著名キャンペーンにおいては)**様々な工夫を仕込んだ上で公開される**

例えば、安定稼働のためJSランタイムのBunを取得する

```
{
  "name": "keyv",
  "version": "6.0.0",
  "scripts": {
    "preinstall": "node setup.mjs"
  },
  "files": ["dist", "setup.mjs",
"Math_Symbol.js"]
}
```

1. npm install が setup.mjs を起動する
2. Bun が無ければ GitHub Releases から取得する
3. bun Math_Symbol.js を別プロセスで起動する
4. CI 以外では親から切り離して常駐する
5. 200 超のパスを探索し、C2 と GitHub へ送る

- dist/ のライブラリ本体は無改変のため、パッケージは普段どおり動き続けた
- 実行の記録は、ローダから子プロセスへ処理が移るところまで追う必要がある

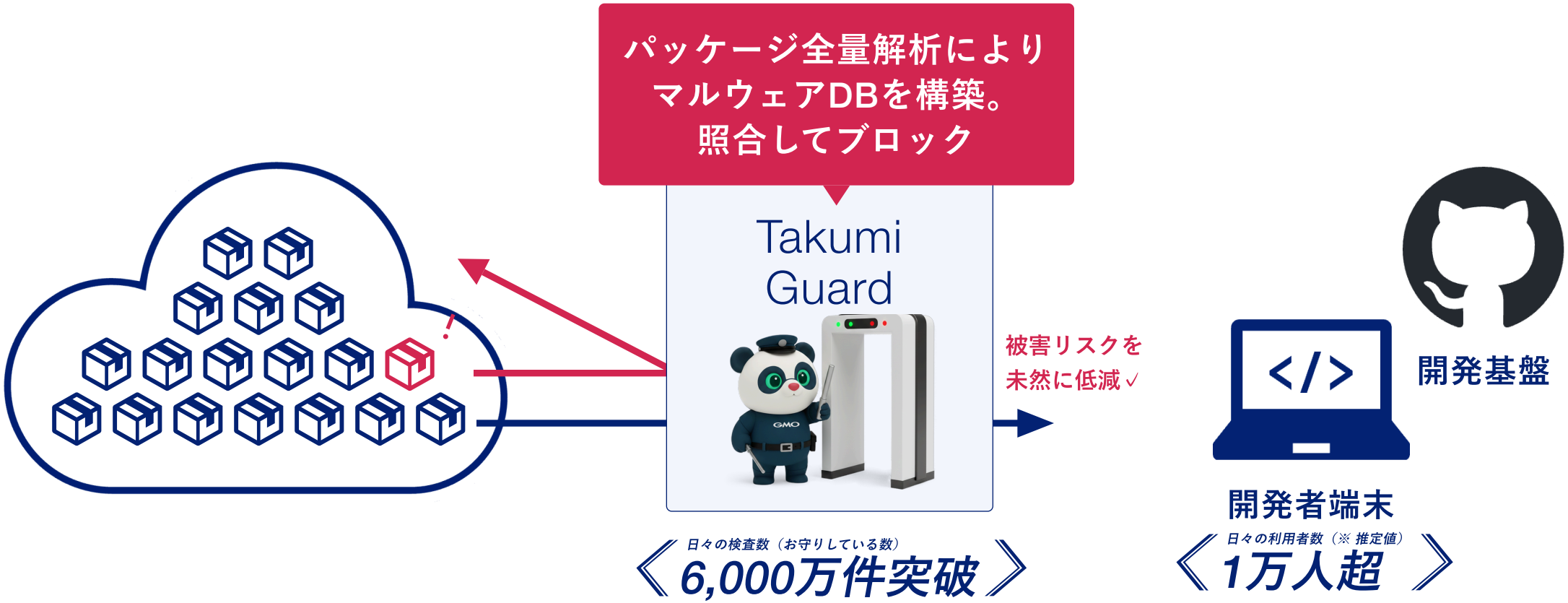
- **いたちごっこである**
  - 攻撃者は小賢しい工夫を都度加えてくる
- **何より、公開経路の検証も条件の削減も悪性コードそのものは見分けない**
  - 我々が Takumi Guard を提供する理由

PART 2

報道のない日

悪性パッケージは毎日公開されていた

Takumi の Guard 機能は、エンジニア向けマルウェアを自動ブロック。
法人なら ①法人への一括導入機能・②感染時の管理者通知も利用可能



半年間に公開されたパッケージを全量解析して確かめた

解析期間：2026年2月19日〜8月18日（181日間、6レジストリ）

19,213,180検体

観測期間中に解析したパッケージ名＋バージョン

31,840検体（0.2%）

期間内に公開され、悪性と判定したもの

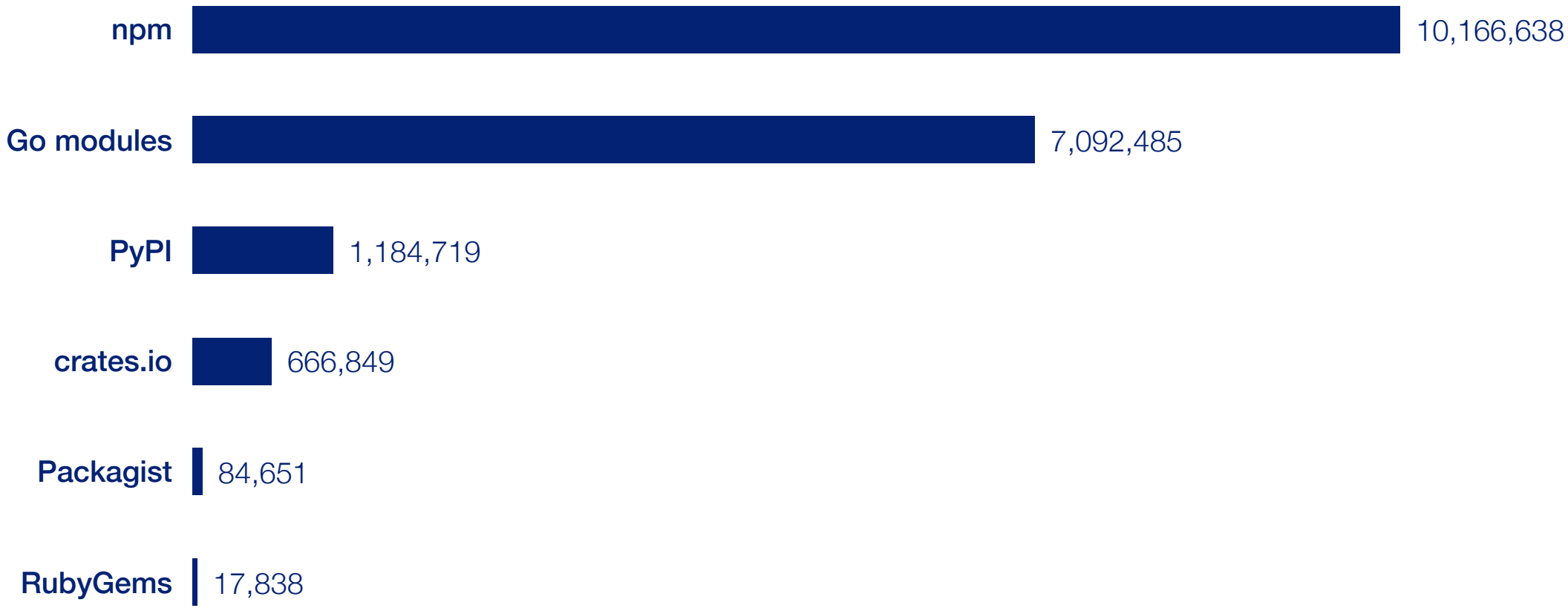
公開直後に取得し、静的解析とサンドボックスでの実行で判定する

1. レジストリの公開イベントを受け取る
2. メタデータとコードを静的に解析する
3. サンドボックスで install と import を実行する
4. 通信先、ファイル、コマンドから判定する

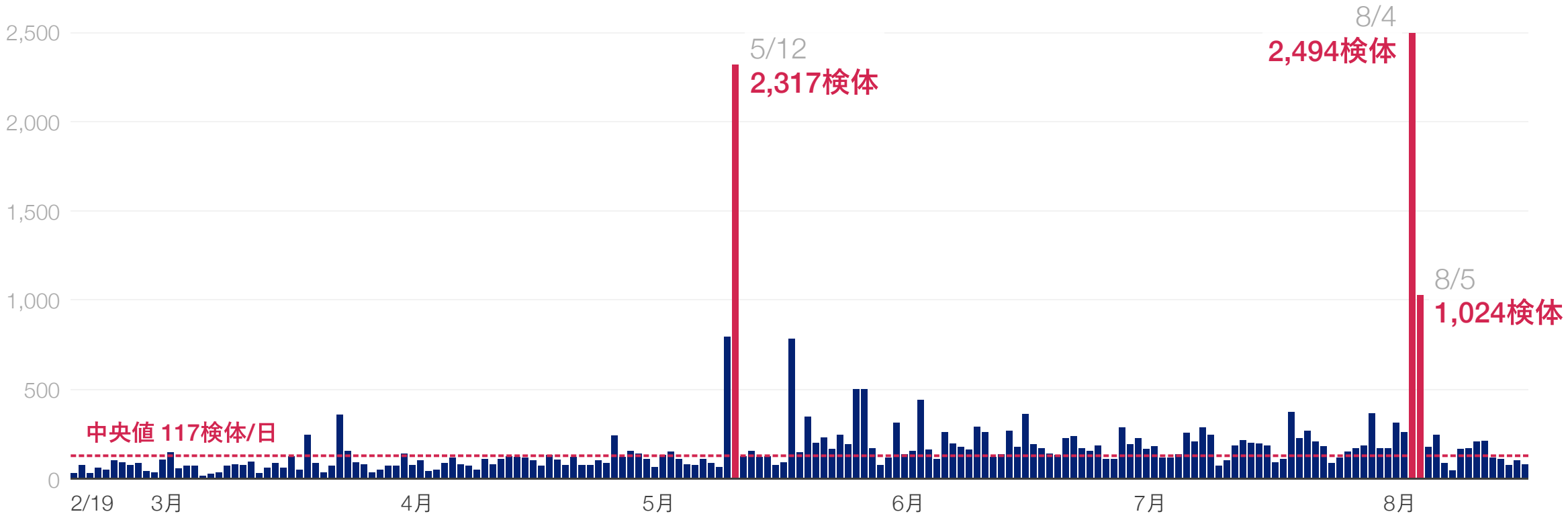
- 削除されると検体も C2 も消えるため、公開直後に取得して保存する必要がある

6レジストリで公開されたバージョンを、公開直後から解析している

エコシステム別の解析検体数



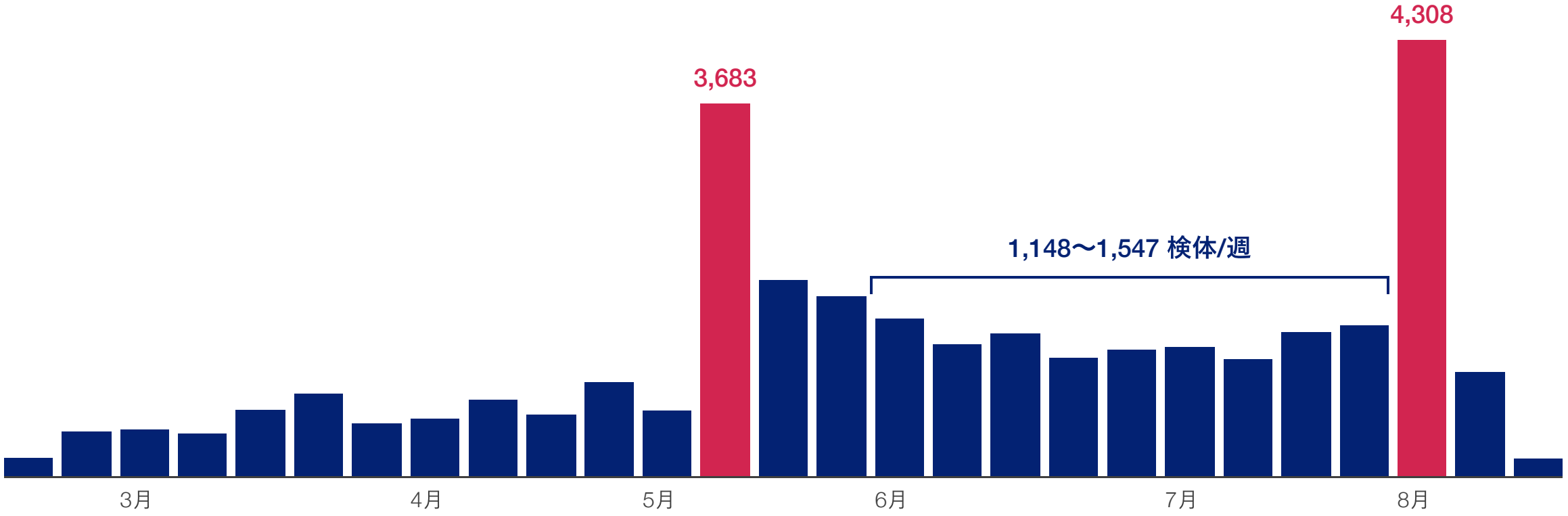
181日すべての日に悪性検体の公開があり、1日の中央値は117検体だった



- 最も少ない日でも 8 検体（3/7）が公開され、ゼロの日はなかった
- 1日の最大は 8/4 の 2,494 検体で、keyv 関連の再公開が重なった

日別の悪性検体数（UTC）

週ごとの山は報道と重なり、報道の少ない週にも千件規模の公開が続いた



- 5/11週は RubyGems の量産型 2,531 検体と、Mini Shai-Hulud の npm 検体が重なった
- 8/3週は keyv 関連の再公開と、新しい名前のクレデンシャル窃取型 npm 検体が重なった

一部のエコシステムは期間の途中から収集を始めたため、前半の水位は低く出ている

PART 3

届き方と狙い

型と活動群から読み解く

通信先を分析すると、攻撃に関わる295の活動群にまとまった

6,387
関連付けの情報が
揃った検体

342
通信先で
連結した群

調査・検証目的の公開
（推定）を除外

295
**攻撃に関わる
活動群**

- C2 やデータ送信先を共有する検体を、推移的に同じ群とした
- 汎用の共有サービスは連結に使わなかった
- 公開直後に削除された / 解析環境を検知した検体は含まれない
- 群の数を攻撃者の数として読むことはできない

活動群の規模と継続期間はばらばらで、数か月続く群もあった

群	継続日数	検体数	公開者数
G1	56日	1,875	18
G5	29日	137	4
G6	2日	98	5
G7	29日	92	1
G8	84日	73	62
G9	44日	63	30
G10	1日	61	4
G11	1日	55	1
G12	9日	55	1
G13	9日	49	4
G14	40日	46	2
G15	21日	39	2
G16	45日	37	15
G17	57日	37	18
G18	48日	36	2
G19	29日	36	5

出典: ソフトウェアサプライチェーン 脅威レポート 2026 図5。期間は各群の検体を最初に記録した日から最後に記録した日まで

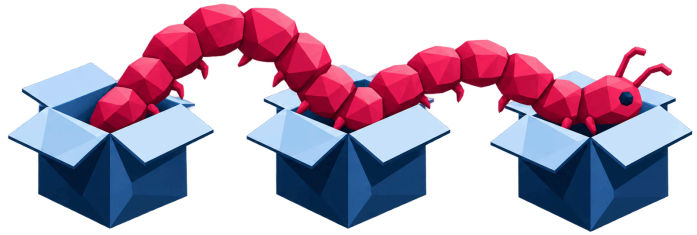
悪性パッケージは3つに分類できる



著名侵害型

正規の公開者や CI を奪い、既存パッケージに悪性版を出す。遠隔操作用の RAT が多い

例: axios、Mastra



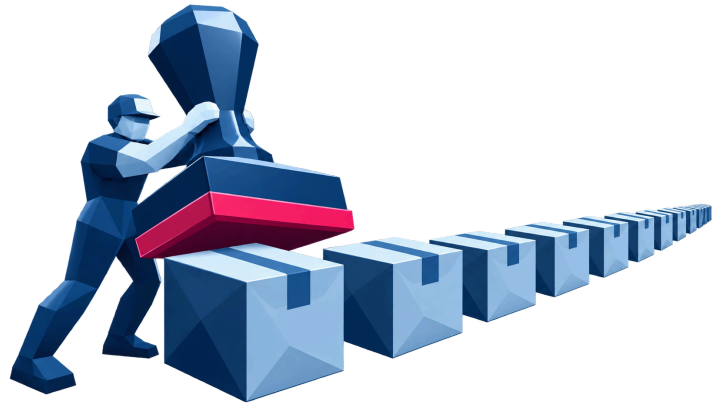
ワーム型

感染先で得た公開クレデンシャルで、別の既存パッケージを再公開する。情報窃取が中心

例: Mini Shai-Hulud、keyv

正規の公開経路を悪用

いつもの更新で届く



量産型

新しい名前で繰り返し公開する。追加のコードを取ってくるドロッパーが多い

例: 偽装名、AI が誤って提案しがちな名前

新しいパッケージを量産

依存を選ぶときに届く

正規ライブラリ、AI ツール、暗号資産ツールを装い、公開の仕方を使い分けていた

群	装った対象と名前の例	公開の仕方
G5	dayjs を複製した easy-day-js、caspian-day-js	正規と同じ 1.11.22 を名乗り、postinstall で外部からコードを取得
G6	crypto-credential-scanner、wallet-security-checker	5公開者が npm と PyPI に同じ窃取処理を移植（98検体）
G7	ai-sdk-helpers、anthropic-toolkit、mcp-server-pg	1公開者が29日間で5つの名前に92検体を重ねた
G8	chai-as-built、chai-as-staged、chai-assertions-plus	62公開者が84日間にわたり73検体を公開
G10	@years18/n8n-nodes-utils-helper-a から末尾だけ変えた名前	4公開者が1日で61検体を、すべて 1.0.0 で公開

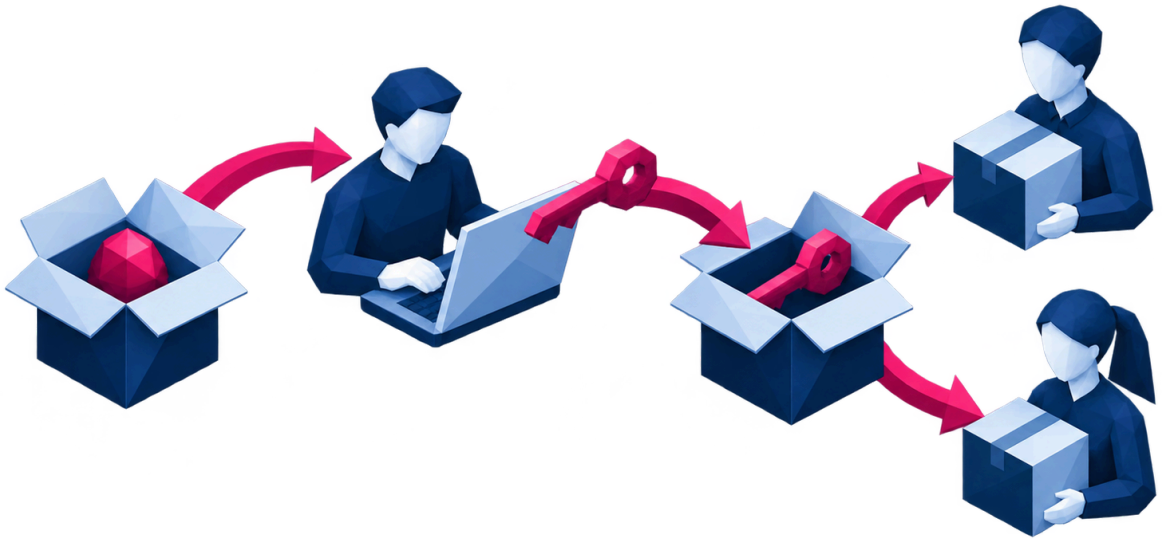
- G5 の easy-day-js は、6月の Mastra 侵害で 141 超のパッケージに依存として注入された

slopsquatting: AI が誤って提案しがちな名前を先回りして登録する

AI が実在しない名前を提案 → 攻撃者がその名前で先に公開 → 提案どおりにインストール

- typosquatting はタイプミスを待つが、slopsquatting は AI の提案を待つ
- AI エージェントが依存を自動で追加する環境では、人の確認を経ずに入りうる

盗んだ公開クレデンシャルで次の悪性版を自動で作り、感染した端末が次の配布元になる



1. ~/.npmrc や gh auth token からクレデンシャルを取る
2. そのクレデンシャルで更新できるパッケージを列挙する
3. 最新の tarball を取得する
4. preinstall とローダを追加する
5. パッチ番号を1つ上げる
6. 正規の名前で再公開する

【G1】1つの公開者アカウントの侵害が、392の既存パッケージに広がった

1 アカウント

最初に侵害された公開者

18 公開者

ワームが盗んだ鍵で連鎖した正規の公開者

392 パッケージ

悪性版が出た既存パッケージ

うち週間ダウンロード上位

keyv	約1億9,700万回
file-entry-cache	約1億9,500万回
cacheable-request	約5,000万回

いつもの更新と推移的依存で届く

活動群 G1 の観測値。ダウンロード数は全バージョン合計（npm Downloads API）

開発端末と CI は、組織の鍵束と開発者個人の財布を同時に持っている

組織の権限	
ソース管理・公開	gh auth token、~/.npmrc、~/.pypirc
クラウド	~/.aws/credentials、全リージョンの Secrets Manager
コンテナ	~/.kube/config、~/.docker/config.json
秘密管理	~/.vault-token、.env、シェル履歴
CI	Actions ランナーのプロセスメモリ

個人の資産と AI 開発ツール	
ブラウザ	保存パスワード、セッション、Keychain
暗号資産	ウォレットの秘密鍵とシード
メッセージ	Slack、Discord、Signal のローカルデータ
AI ツール	~/.claude/、MCP 設定、Cursor
設定改変	.claude/settings.json で次回起動時に再実行

出典: 脅威レポート 2026 表2（Mini Shai-Hulud、@bitwarden/cli、keyv 関連活動などの解析結果）

対岸の火事ではない3つの理由

理由	観測から分かったこと
毎日公開されている	181日すべての日に悪性検体が公開され、中央値は1日117検体だった
影響が爆発的に広がる	著名侵害型とワーム型が既存パッケージを汚し、推移的依存でいつもの更新に乗って届く
狙われる理由がある	開発端末と CI には、クラウドや公開の鍵、個人の資産、AI ツールの設定が集まっている

狙われるかどうかではなく、配布の経路上にいるかどうか

PART 4

10月の3事案

攻撃は部品化して続いている

10月にはすでに3件著名パッケージの侵害が発生

2月 3月 4月 5月 6月 7月 8月 9月 10月

2月〜3月
Trivy
Action のタグと公式リリースを改変

3/24・3/27
LiteLLM・Telnyx
Trivy で漏れたクレデンシャルが起点とみられる

3/31
axios
週1億 DL 超。依存1行の追加で RAT

4/23
@bitwarden/cli
公開用ワークフローを改変

4/29〜5/11
Mini Shai-Hulud・TanStack
ワームが npm と PyPI に拡散

5月
Laravel-Lang
Composer の autoload.files で発火

6/17
Mastra
依存注入

7/14
AsyncAPI
import 時の発火

8/4
keyv
週1億 DL 超。ワームで多数を再公開

10/5
@subql/common
正規の Trusted Publisher から悪性版

10/8
tensorlake
keyv と同系統。失効で rm -rf ~/

10/21
LiteLLM 再侵害
ソースはクリーン、wheel だけ汚染

3月事案と異なる原因で、誤って公開されたシークレットの放置による

2026-05-28

管理キーを誤公開

Agent Platformの管理キー。
約3時間後にファイルは削除。
公開Git履歴には残り、
キーの失効も未実施。

→

2026-10-11

残ったキーで内部へ侵入

APIから組織Ownerの
GitHubトークンを取得。
CI経由で別のトークンも回収。

→

2026-10-21

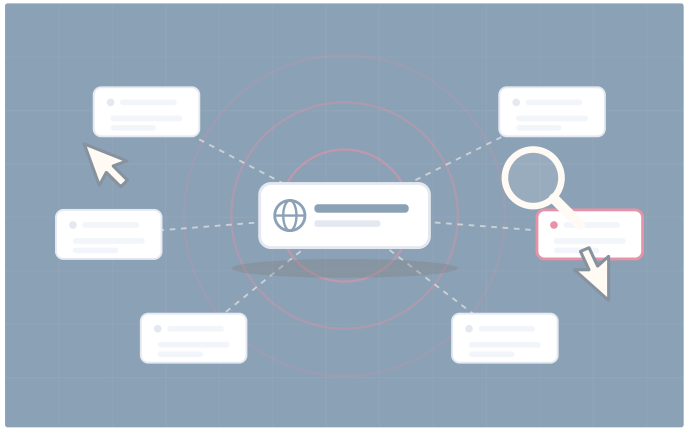
公開ワークフローを改ざん

Owner権限で保護を上書き。
別リポジトリからビルドし、
悪性v1.107.0を正規経路で公開。

漏洩キーの失効と、内部サービスへ渡す権限の最小化が必要

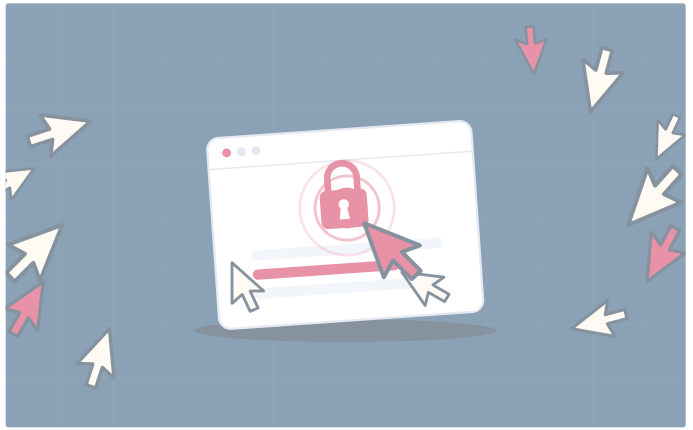
出典：LiteLLM公式RCA「The attack chain behind the v1.107.0 LiteLLM release」（2026-10-23公開）
日付は日本時間。2026-11-15確認。

攻撃者目線で、資産探索から侵入まで一気通貫。機微情報の漏洩も調査



資産探索

ドメインを手がかりに、攻撃者と同じ方法でインターネット上の公開資産を洗い出します。



ペネトレーションテスト

見つけた資産すべてを対象に、AIが攻撃者と同じ手口で実際に侵入を試みます。



機微情報の漏洩調査

GitHubなど、インターネット上の公開情報から、お客様に関係するパスワードやAPIキーなどが漏れていないかを調べます。

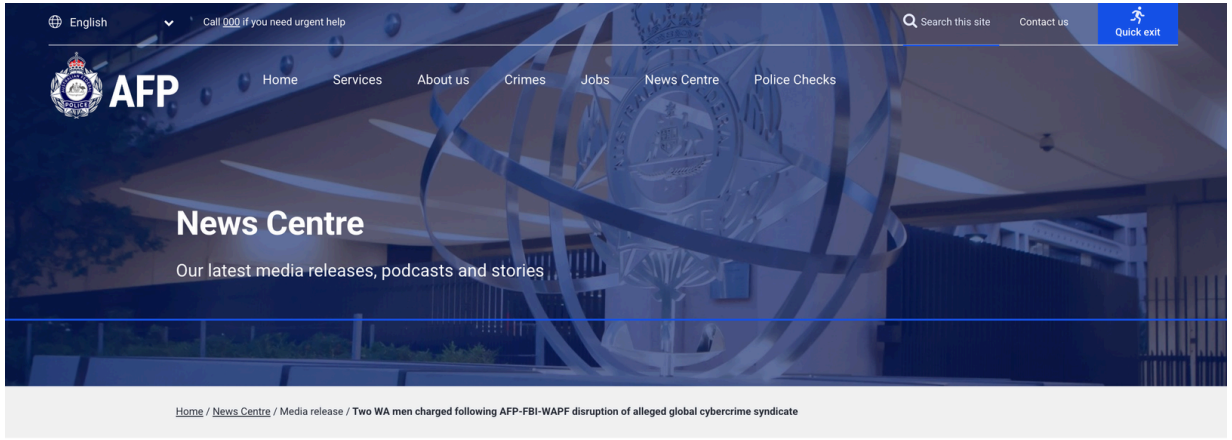
subql・tensorlakeはオープンソース化された Shai-Huludの悪用と推測される

2025年9月	2025年11月	2026年4月〜5月	2026年5月	2026年8月	2026年10月
初代 Shai-Hulud	Second Coming	Mini Shai-Hulud	ソースの公開	keyv	@subql・tensorlake

- 初代は集めた情報を webhook.site に送るだけの素朴なスクリプトだった
- 1年で、暗号化、Bun での実行、永続化、ワームによる拡散、自爆まで揃った
- 2026年5月にソースが公開され、**誰でも部品として使える**ようになった

出典: Socket「Happy Birthday, Shai-Hulud」（2026-09-18）（https://socket.dev/blog/happy-birthday-shai-hulud）、当社ブログ各記事

2026年8月、豪州で TeamPCP のメンバー2名が逮捕・起訴されているが ...



Two WA men charged following AFP-FBI-WAPF disruption of alleged global cybercrime syndicate

出典: Australian Federal Police メディアリリース（2026-08-27）

10月の3事案も、単独の対策の抜け道を突く"工夫"がある

対策	@subql/common	tensorlake	LiteLLM 再侵害
Provenance の確認	**効かない** 正規の OIDC 経路で公開	**効かない** 有効な Provenance 付き	**判定に使えない** 正規版にも Provenance なし
インストール時実行無効化（--ignore-scripts等）	**効かない** **require** 時にも発火	**効いた** 発火は **preinstall** のみ	**効かない** **.pth** で起動時に発火
GitHub のソースを確認する	**効かない** CI の中で tarball を差し替え	**効いた** 悪性ファイルは **main** にも存在	**効かない** タグはクリーン、wheel だけ汚染
初動で鍵を先に失効させる	**効いた**	**逆効果** 失効を検知して **rm -rf ~/**	**効いた** 常駐デーモンを止めてから
公開直後を待つ（7日）	**効いた** 約50分で削除	**効いた** 約50分で削除	**効いた** 同日中に削除

- 3事案とも、試験版の公開や事前の証明書取得など、公開前に準備を重ねていた

クレデンシャルの失効が、破壊の引き金になる悪質な設定も

通常問題ない初動

1 侵害を知り、GitHub トークンを失効させる

2 常駐する gh-token-monitor が60秒以内に失効を検知

3 rm -rf ~/ でホームディレクトリが消える

- 同じ挙動は5月の Mini Shai-Hulud 第2波でも確認され、個別の失効手順が必要になっている

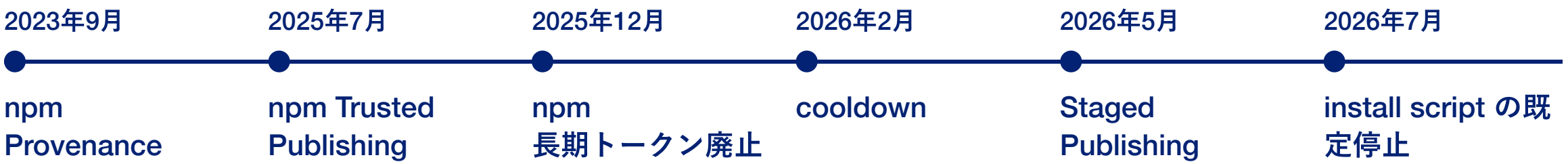
PART 5

自社を守るヒント

止められない前提で層を重ねる

エコシステムの対策は進んだが、正規の経路から出る悪性版は止められない

npm の例



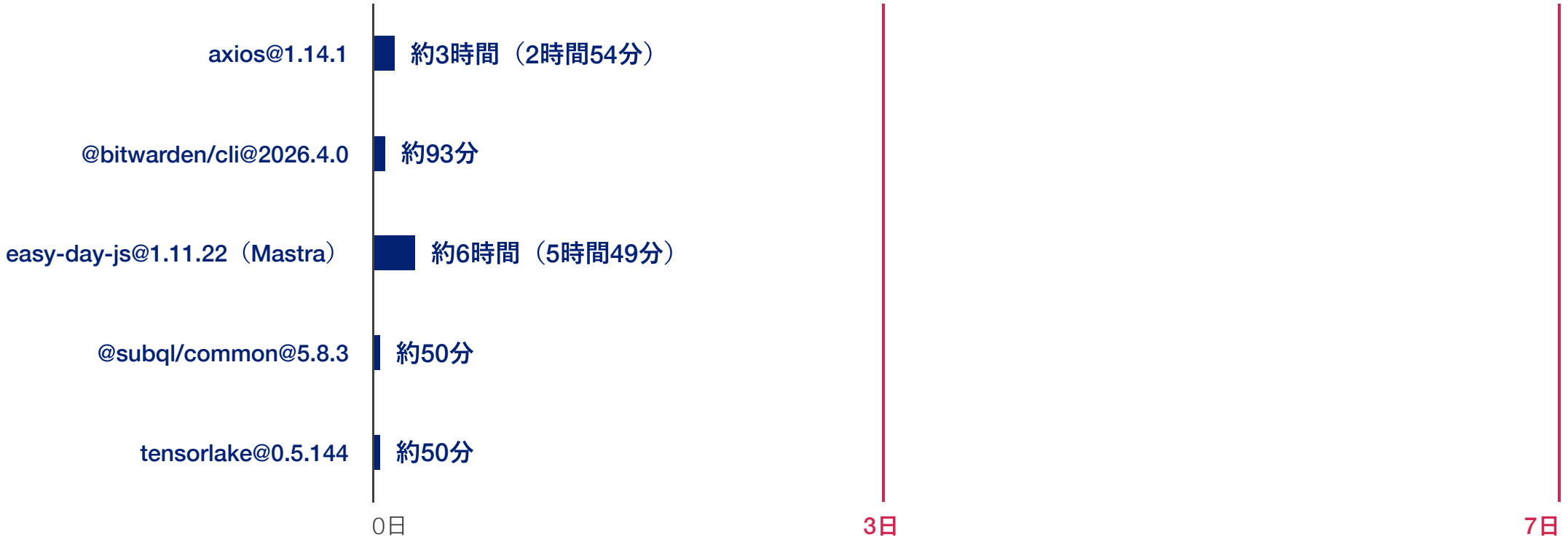
- @bitwarden/cli、TanStack、keyv の悪性版など、**有効な真正性の検証を通過するものが存在**
- npm の高影響パッケージのうち、Staged Publishing まで導入したのは約17%にとどまる

出典: sxzz/npm-top-publishing（https://github.com/sxzz/npm-top-publishing）（2026年10月時点）

「導入の防止」「感染可能性の把握」「影響の限定」の3層を重ねる

層	主な手段
導入の防止	・既知の悪性バージョンを導入前に遮断する（**Takumi Guard**） ・公開直後の更新を待機させる ・バージョンを固定し、意図しない変更を防ぐ
感染可能性の把握	・取得履歴を残し、後から悪性と判明したら通知を受ける（**Takumi Guard**） ・開発端末の EDR でプロセスと通信を記録する ・CI を隔離環境で実行し、記録から検知する（**Takumi Runner**）
影響の限定	・依存の導入とビルドを、重要な権限から切り離す ・強い鍵を置かず、短命なクレデンシャルを使う ・外向き通信を必要な宛先に絞る

cooldown: 数日待つだけで、話題になる事案を避けられる



- Dependabot の cooldown や Renovate の minimumReleaseAge で、公開直後のバージョンを自動更新から一定期間外す
- 待機期間のうちに、レジストリによる削除とセキュリティ調査が進む
- 脆弱性修正は待機の対象から外し、**通常の更新と緊急更新を分ける**

取得するバージョンを固定し、意図しない変更を防ぐ

1

依存を固定する

直接依存は狭いバージョン範囲を指定し、解決結果をロックファイルとダイジェストで固定する

2

Actions も固定する

GitHub Actions はタグではなくコミットハッシュで参照する

3

更新は差分で確認する

更新は自動 PR に集約し、名前・バージョン・取得元・追加スクリプトを1つの差分で確認する

（こちらも）固定したままでは脆弱性修正の導入が遅れる。待機期間付きの自動更新と組み合わせる

Takumi の Guard 機能は、エンジニア向けマルウェアを自動ブロック。
法人なら ①法人への一括導入機能・②感染時の管理者通知も利用可能

パッケージ全量解析により
マルウェアDBを構築。
照合してブロック



日々の検査数（お守りしている数）
6,000万件突破

日々の利用者数（※ 推定値）
1万人超

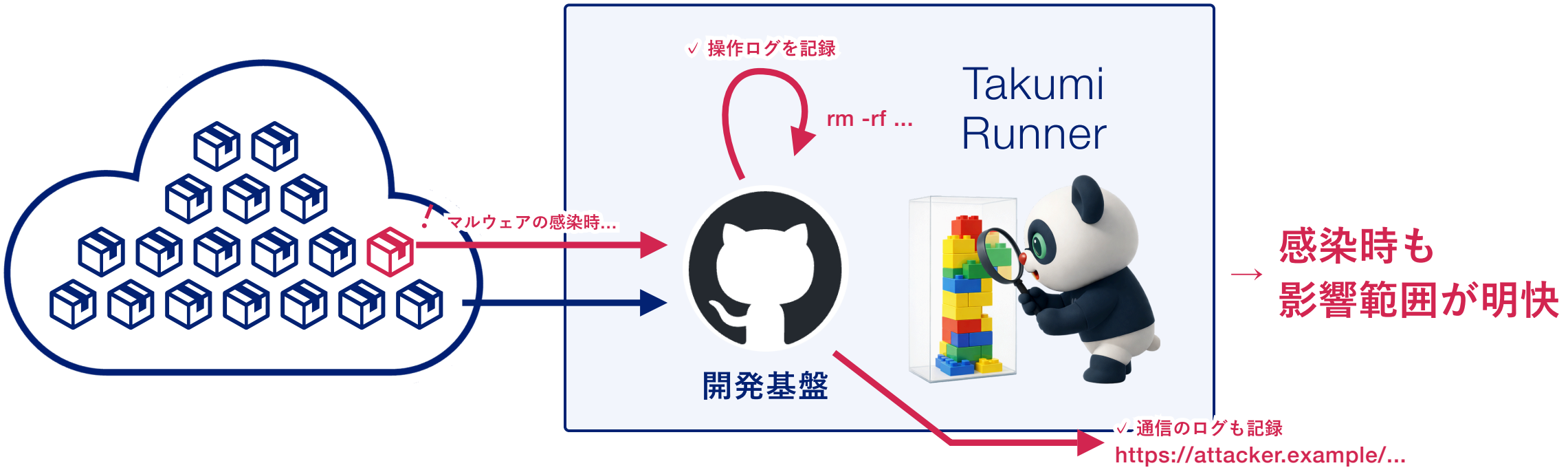
「報じられた朝、自社はそのバージョンを取り込んだか」に即答できるか



- 取得履歴を残し、後から悪性と判明したら通知を受ける（**Takumi Guard**）
- 開発端末の動きを EDR で記録する
- CI の動きを隔離環境で記録する（**Takumi Runner**）

Takumi Runner は、GitHub Actions などの CI/CD 環境（開発基盤 ※）での感染時に影響調査・事後対応を助ける 機能。万が一に備える

※ 既存アンチウイルス・EDR 等では対象外の傾向。侵害時、通常は影響調査が困難



依存の導入とビルドを、重要な権限から切り離す

隔離環境（コンテナ・CI）

- SSH 鍵、公開権限、本番のクラウド権限を置かない
- クレデンシャルや Docker ソケットを共有せず、必要なファイルだけ読み取り専用で渡す
- 公開ジョブを分離し、トークンは対象・権限・有効時間を絞って渡す
- 外向き通信は必要な宛先に絞り、許可と遮断を記録する

開発端末

- IDE や AI ツールまですべて隔離するのは現実的でない
- 当面は、重要なクレデンシャルを隔離環境に渡さず、共有する経路を EDR で監視する
- 今後は、必要な操作のときだけ短命なクレデンシャルを渡す仕組みが要る

悪性パッケージの公開は毎日。脆弱性と両輪で向き合う必要がある

半年間の観測では、181日すべての日に悪性検体が公開された

1日2,500検体に迫る大規模な侵害も。推移的依存も含め、対岸の火事ではない

11月にはGitHub Actionsも標的に。盗まれた権限が次の侵害の足掛かりになる

対策として、入れない、気づく、影響を限定するの3層を重ねる必要がある

ニュースのない日にこそ、次の侵害への備えを進めてほしい


GMO Flatt Security
エンジニアの背中を預かる
より多くのエンジニアがものづくりに集中できる社会を、
セキュリティ面からつくる会社