DSCI
PROMOTING DATA PROTECTION
A nasscom Initiative

DSCI THREAT INTELLIGENCE AND RESEARCH INITIATIVE

THREAT REPORT

MAY 2026

TABLE OF CONTENTS


MIASMA CAMPAIGN
REDHAT NPM PACKAGES
COMPROMISED

Miasma Campaign: Redhat npm Package Compromised

Introduction

Multiple npm packages published under the @redhat-cloud-services namespace were compromised as part of a supply-chain attack dubbed Miasma, a variant or derivative of the previously documented Mini Shai-Hulud malware family. The attack leveraged trusted software distribution channels to distribute credential-stealing malware through legitimate Red Hat npm packages.

Unlike traditional package hijacking incidents involving stolen maintainer credentials, evidence suggests the threat actor abused GitHub Actions Trusted Publishing mechanisms and legitimate CI/CD workflows to publish malicious packages with valid provenance signatures. This significantly increased trustworthiness and reduced the likelihood of detection by users relying on provenance verification.

The campaign, therefore, represents a high-severity software supply-chain compromise affecting developers, build systems, and enterprise cloud environments.

Incident Overview

The malware executed automatically during package installation through a malicious preinstall script and targeted:

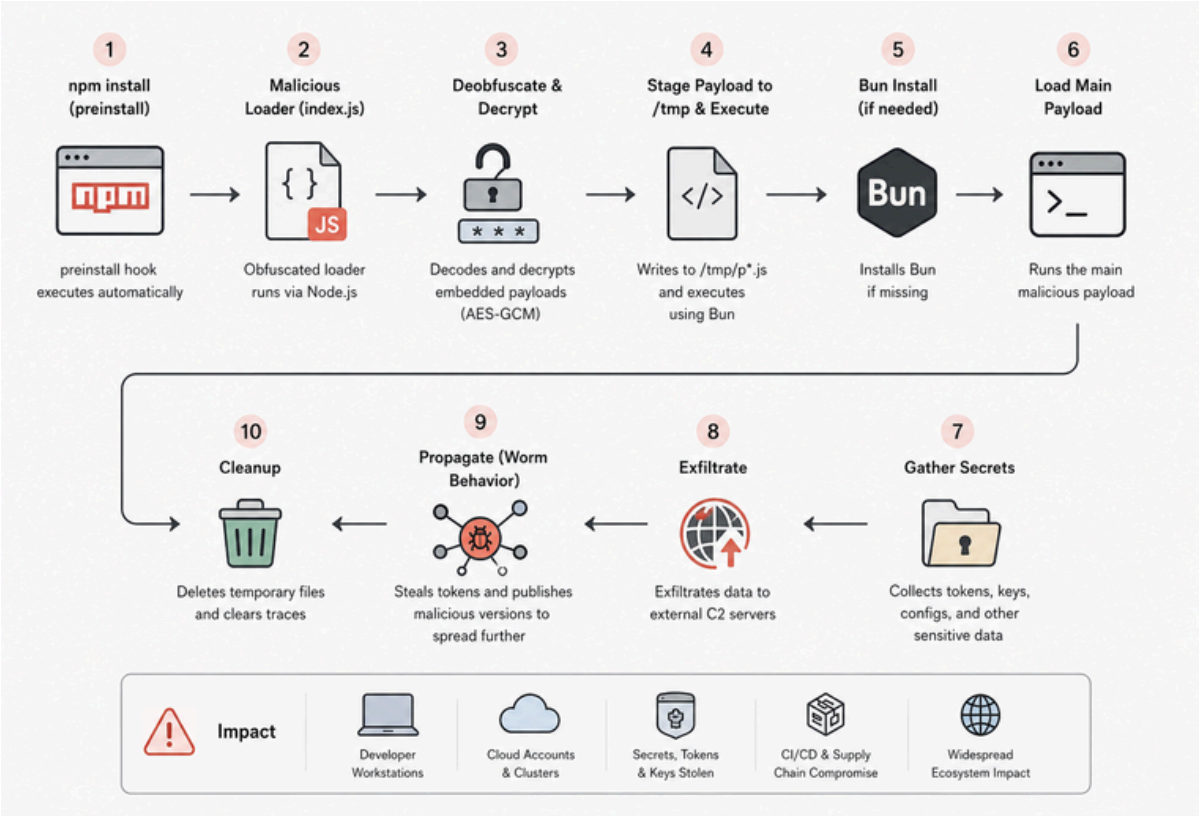
- GitHub Actions secrets, CLI tokens
- npm authentication tokens
- AWS, Azure, and GCP credentials and metadata
- Kubernetes configurations
- HashiCorp vault credentials
- SSH private keys
- Git credentials
- Password manager secrets
- CI/CD pipeline environments
- .env, .netrc, .pypirc files
- Cryptocurrency wallet files

Field	Details
Campaign Name	Miasma
Related Malware	Mini Shai-Hulud
Target	npm ecosystem
Affected Scope	@redhat-cloud-services
Primary Objective	Credential theft and supply-chainpropagation
Initial Access	GitHub Actions Trusted Publishing abuse
Malware Type	JavaScript credential stealer/worm
Execution Method	npm preinstall hook
Potential Impact	CI/CD compromise, cloud account takeover, software supply-chain infection

Attack Chain Analysis

Packages	Versions
@redhat-cloud-services/vulnerabilities-client	2.1.11, 2.1.9
@redhat-cloud-services/topological-inventory-client	3.0.13, 3.0.11
@redhat-cloud-services/rule-components	4.7.5, 4.7.3
@redhat-cloud-services/rbac-client	9.0.6, 9.0.4
@redhat-cloud-services/quickstarts-client	4.0.14, 4.0.12
@redhat-cloud-services/patch-client	4.0.7, 4.0.5
@redhat-cloud-services/integrations-client	6.0.7, 6.0.5
@redhat-cloud-services/frontend-components-utilities	7.4.4, 7.4.2

Attack Chain Analysis



*This flowchart was generated by AI and reviewed for accuracy.**

Phase 1: Trusted Publishing Abuse

The attacker allegedly abused GitHub Actions Trusted Publishing workflows. Traditional npm attack usually requires stolen npm tokens and compromised maintainer accounts, but in this case:

- No maintainer account compromise was observed.
- No npm token theft appears.
- Packages were published using legitimate Red Hat CI/CD identities.

Researchers observed that there was:

- Creation of temporary branches.
- Modification of GitHub Actions workflows.
- Abuse of GitHub Open ID Connect (OIDC) identity tokens.
- Generation of valid npm publish permissions.
- Publication of trojan package versions.

The attack bypasses many traditional controls like MFA, token rotation, publisher verification, provenance validation, etc. because malicious packages were produced by legitimate publishing pipelines.

Phase 2: Malicious Package Distribution

Step 1: npm install

The threat actor inserted *{"preinstall": "node index.js"}* or equivalent execution logic in the package.json file.

This causes the malware execution immediately during *"npm install"*

- No application execution is required
- No import statement is required
- No developer interaction beyond package installation is required

Step 2: Malicious Entry Point

A legitimate package looks like *{"main": "esm/index.js"}* where this specific javascript file appears legitimate and may contain the actual application code. However, the attacker configured

```
{
  "main" : "index.js"
  "module" : "esm/index.js"
}
```

Security reviewers may inspect *"esm/index.js"* because it looks like it contains the malicious logic but node.js, npm lifecycle scripts, commonJS, etc. generally uses *"main"*, so the malicious file gets executed.

Step 3: Obfuscated Loader

The *"index.js"* contains *[40,120,112,...]* which means that it used ROT-3 (Caesar Cipher Shift) which shifts letters by 3 position, so

- *D = A*
- *E = B*
- *F = C*

Basically, this malware converts numbers into characters, shift characters and execute it using *eval()*. After deobfuscation, the malware produces an async JavaScript wrapper which imports crypto APIs and decrypt the payloads.

Step 4: AES_GCM Payload Decryption

This wrapper contains encryption blobs such as

createDecipheriv(
“aes-128-gcm”
.....
)

It converts large binary objects into unreadable ciphertext due to which the real malware remains hidden and security tools will not flag it.

It decrypts 2 payloads, in which

- Payload 1 (_b) - Checks if Bun exist or not and install it, if necessary.
- Payload 1 (_p) - Main malware.

Step 5: Temporay File

The malware then generates a *“/tmp/p"+random+".js”* file to evade detection rules. The payload is then written to disk so that it is ready for execution. This is the first time when actual malware is available in a readable form as before this step, it remains encrypted.

Step 6: Execute Payload

The malware is executed using *“bun run payload.js”* instead of *“node payload.js”* to reduce detection and fast execution. The malware also verifies if bun is installed or not, if not then it uses *“_b”* payload to install bun and execute the malware.

Step 7: Runtime String Decryption

The main payload uses a large string table and a custom decryption function. Even after the payload is decrypted, the strings remain hidden as the malware uses *f4abccab2()* to decrypt individual strings and the decoded values contain:

- https://api.github.com
- api.anthropic.com
- python-requests/2.31.0
- GitHub Commands

- IfYouInvalidateThisTokenItWillNukeTheComputerOfTheOwner
- thebeautifulmarchoftime

These strings are only revealed during execution.

Step 8: Reconnaissance & Credential Harvesting

Using the malware it collects different data such as tokens, keys and credentials related to GitHub, AWS, Azure, GCP, npm, Docker, Kubernetes account tokens, SSH private keys, .env file, cryptocurrency wallet files, hotnames, usernames, etc.

After execution it removes the *"/tmp/p*"* files to reduce any forensic evidence.

Comments

There were a lot of inline comments in the payloads and packages to describe the malware functionality such as:

- *{*
  "preinstall" : "node index.js"
  //Executes index.js automatically during npm install
  }
- *"//Decrypts embedded runtime payloads that are hidden from static/package review"*
- *"//Silently downloads an execution runtime during package install"*
- *"//Avoids execution or chnages behaviour on Russian-language systems"*
- *"// Collects the local GitHub CLI authentication token"*
- *"//Targets cloud credentials, SSH private keys, Git credentials, cryptocurrency wallet files"*

MITRE ATT&CK Mapping

Tactic	Technique (ID)	Technique (Name)	Procedure
Initial Access	T1195	Supply Chain Compromise	Trojan versions of legitimate @redhat-cloud-services npm packages
	T1199	Trusted Relationship Abuse	Malicious packages published through trusted package distribution channels
Execution	T1059.007	Command and Scripting Interpreter (JavaScript)	Malicious *index.js* loader executed JavaScript code through eval()
	T1204	User Execution via Package Installation	Automatic malware execution when *“npm install”* is used
Persistence	T1556	CI/CD Workflow Modification	Abuse or modification of GitHub Actions Trusted Publishing workflows
Defense Evasion	T1027	Obfuscated Files or Information	Payload employed character arrays, Caesar/ROT transformations, etc.
	T1140	Deobfuscate/Decode Files or Information	Used AES-GCM decryption to decrypt payloads

Tactic	Technique (ID)	Technique (Name)	Procedure
Discovery	T1082	System Information Discovery	Malware enumerates host information, installed software etc.
	T1083	Environment Variable Discovery	Inspected environment variables for secrets, API tokens, cloud credentials
Credential Access	T1555	Credentials from Password Stores	Targeted credentials from password managers and authentication utilities
	T1552.001	Credentials from Files	Searched local files such as .npmrc, .gitconfig, .env, SSH keys
	T1528	Cloud Credentials	Harvested AWS, Azure, GCP, Kubernetes, etc.
Collection	T1005	Data from Local System	Collected configuration files, authentication tokens, secrets, keys
Exfiltration	T1567	Exfiltration Over Web Services	Stolen credentials and collected data transmitted to attacker-controlled infrastructure
Lateral Movement	T1195	Supply Chain Compromise	Stolen npm and GitHub credentials could be leveraged to publish additional malicious packages

Tactic	Technique (ID)	Technique (Name)	Procedure
Impact	T1496	Resource Hijacking/ Further Compromise	Compromised credentials enabled potential takeover of repositories, CI/CD pipelines, cloud environments, and software supply chains
Collection	T1005	Data from Local System	Collected configuration files, authentication tokens, secrets, keys
Exfiltration	T1567	Exfiltration Over Web Services	Stolen credentials and collected data transmitted to attacker-controlled infrastructure

Indicators of Compromise (IOCs)

SHA-256 Hashes:

- @redhat-cloud-services_chrome-2.3.1.tar.gz: 88896d478986d453f5da79b311de39d9b4b1bea95c21af1d8ef181b0f4e52fe9
- package/index.js: 21b6409a7b84446310daca5409ad6112ac60a1e4bef97736e53fff5f63bfdef4
- package/package.json: ee262510cb246d2b904991aee7fc61162bdae34463439ec6383bd5356479d362
- Decrypted Bun helper: ac2a2208e1726e008be6c73dc0872d9bba163319259dff1b62055ac933ca46b6
- Decrypted main payload: 0dc06ecdaa63fe24859cfd955053c23245c536e4733480239d14bebf12688e35

Token patterns:

- gh[op]_[A-Za-z0-9]{36,}
- npm_[A-Za-z0-9]{36,}
- ghs_\\d+_[A-Za-z0-9_-]+\\.[A-Za-z0-9_-]+\\.[A-Za-z0-9_-]+
- ghs_[A-Za-z0-9]{36,}

String Indicators:

- f4abccab2
- thebeautifulmarchoftime
- IfYouInvalidateThisTokenItWillNukeTheComputerOfTheOwner
- "preinstall":"node index.js"
- "main":"index.js"
- createDecipheriv("aes-128-gcm"
- createCipheriv("aes-256-gcm"
- RSA_PKCS1_OAEP_PADDING
- oaepHash:"sha256"

The complet list of IOCs is available here.

Recommendations

Immediate Actions (0-48 Hours)

- Identify all systems, repositories, CI/CD pipelines, build servers, container images, and developer workstations that installed affected @redhat-cloud-services package versions.

- Isolate potentially affected hosts and suspend impacted CI/CD workflows to prevent further execution of malicious code.

- Remove malicious package versions from projects, clear dependency caches, and rebuild applications from known-clean environments.

- Assume credentials accessible during package installation have been exposed and immediately rotate GitHub tokens, npm tokens, cloud credentials (AWS, Azure, GCP), Kubernetes tokens, Vault secrets, Docker registry credentials, SSH keys, and other CI/CD secrets.

- Preserve logs, build artifacts, and forensic evidence for investigation and determine whether malicious packages were installed on developer systems, GitHub Actions runners, release pipelines, or package publishing workflows.

Short-Term Actions (7-14 Days)

- Conduct a comprehensive review of GitHub repositories, CI/CD pipelines, npm publishing activity, and cloud environments for signs of unauthorized access, workflow modifications, suspicious package releases, or anomalous credential usage.
- Hunt for persistence mechanisms and indicators of compromise, including unauthorized workflow files, modified developer tooling configurations, temporary Bun installations, randomized payload files, and unusual execution chains involving Node.js, Bun, GitHub CLI, or credential-access commands.
- Validate the integrity of source repositories, package dependencies, build artifacts, and deployment pipelines before returning affected systems to production.
- Enhance monitoring and detection capabilities to identify unusual outbound network connections, secret-access activity, unauthorized package publishing, workflow modifications, and suspicious dependency-installation behavior.
- Review all software and release artifacts generated during the exposure period and redeploy trusted versions where necessary.

Long-term Actions (Three-Five Weeks)

- Strengthen software supply-chain security through dependency allowlisting, lockfile enforcement, SBOM generation, software composition analysis (SCA), package verification, and controlled package approval processes.
- Implement least-privilege access controls across CI/CD environments, restrict GitHub Actions permissions, separate build and publishing workflows, and limit exposure of sensitive credentials during dependency installation and build processes.

- Improve secrets management by adopting short-lived credentials, centralized secret storage, regular credential rotation, and minimizing credential availability within development and automation environments.
- Deploy network egress controls, runtime monitoring, and behavioral detection mechanisms capable of identifying credential harvesting, payload staging, unauthorized downloads, and suspicious process execution.
- Establish continuous software supply-chain monitoring and workflow integrity validation, recognizing that trusted publishing and provenance verification alone are insufficient to prevent compromise of legitimate release pipelines.


SHINYHUNTERS
COMPROMISE
CANVAS
PLATFORM

ShinyHunters Compromise Canvas Platform

Introduction

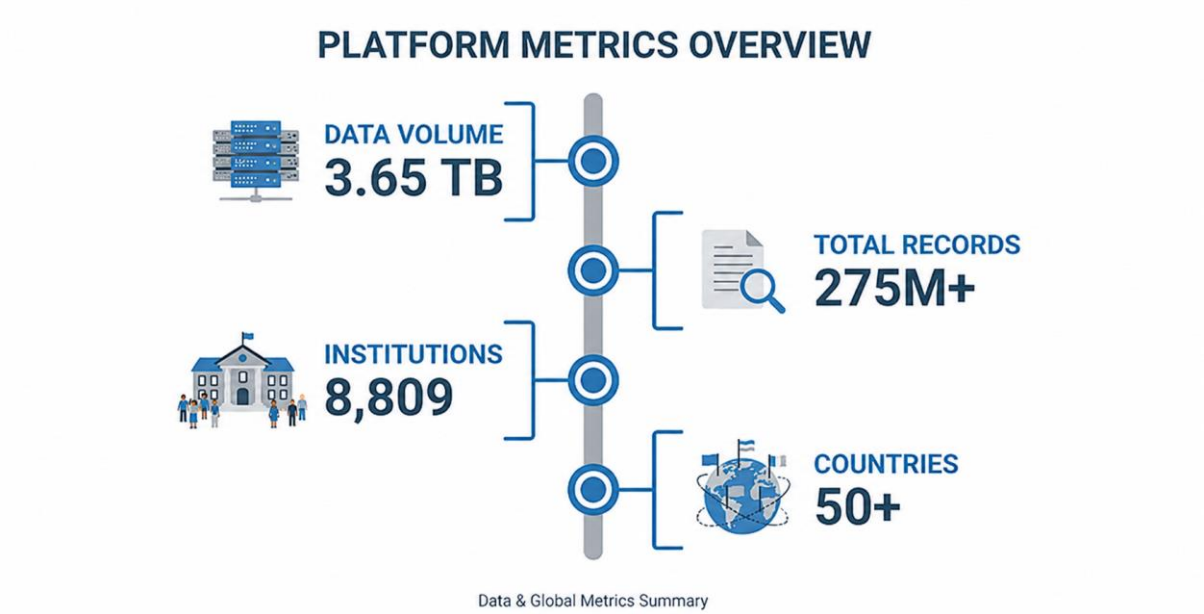
This report provides a comprehensive analysis of the ShinyHunters breach of Instructure's Canvas Learning Management System, discovered on 30 April 2026. The incident represents the largest educational data breach in recorded history, affecting thousands of institutions globally and millions of students, teachers, and staff.

Incident Summary

On 30 April 2026, Instructure Inc., operator of Canvas LMS used by 41% of North American higher education institutions detected unauthorized access to its systems. Threat actors successfully exfiltrated 3.65 terabytes of sensitive educational data containing approximately 275 million records. The breach affected 8,809 educational institutions across 50+ countries, including all eight Ivy League universities, major state universities, and K-12 school systems.

Within days, the threat actor group ShinyHunters publicly claimed responsibility, demanding ransom with a May 12 deadline. When negotiations stalled, the group escalated tactics by defacing Canvas login portals at approximately 330 institutions and implementing direct school-by-school extortion campaigns, leveraging the sensitive nature of stolen student data and threatening permanent publication.

Affected Assets and Users



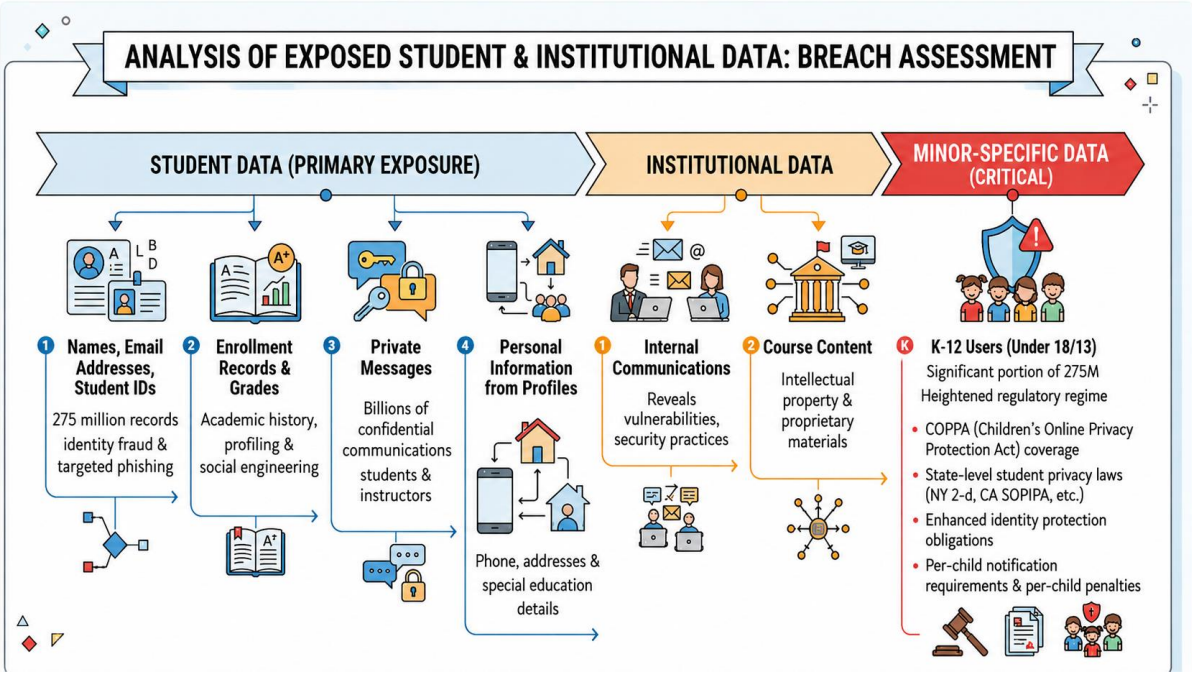
*This flowchart was generated by AI and reviewed for accuracy.**

Threat Overview

Key Impacted Categories:

- **Students:** Names, email addresses, student ID numbers, enrollment records, grades, course activity logs
- **Faculty/Staff:** Contact information, institutional communications, private messages with students
- **Institution-level data:** Internal communications, course content, learning analytics
- **Critical concern:** Significant portion of users are minors (K-12)

Data Exposed

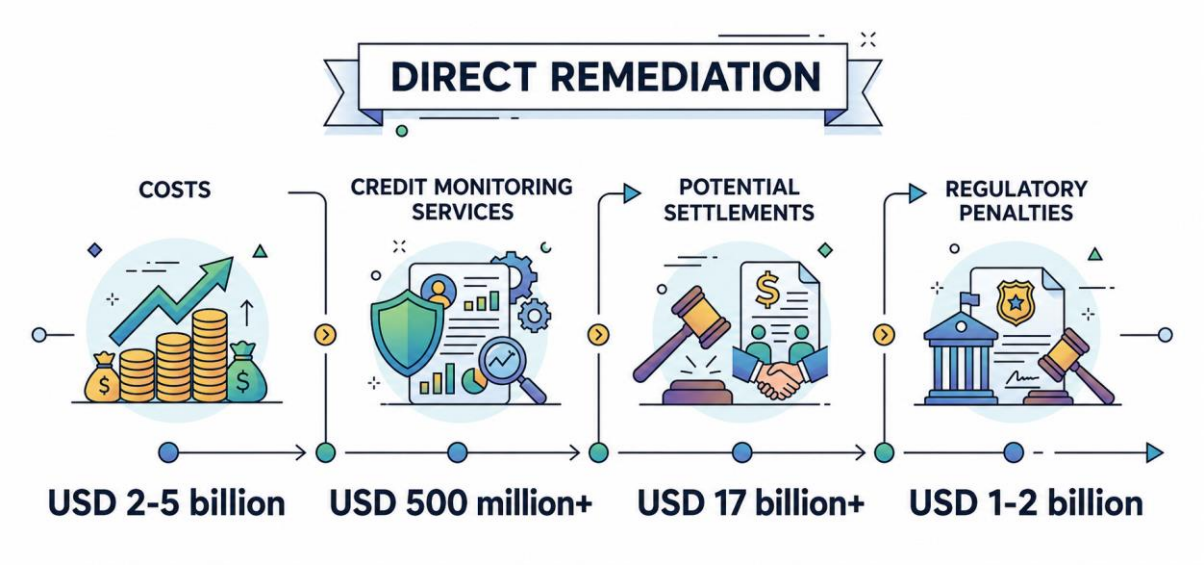


*This flowchart was generated by AI and reviewed for accuracy.**

- **Student Data (275 million records)** — Personal identifiers, academic records, private communications, and profile details including phone numbers, addresses, and special education information, enabling identity fraud, phishing, and social engineering.
- **Institutional Data** — Internal staff communications and proprietary course content revealing security vulnerabilities and intellectual property.
- **Minor-Specific Data (Critical)** — A significant portion of the 275M records belong to children under 18, with millions under 13, triggering heightened legal protections under COPPA, FERPA, and state-level student privacy laws.

Financial Exposure

- **PowerSchool Precedent (Jan 2025)** — A breach of 62 million student records resulted in USD 17.25 million in settlements, 11-state class actions, and more than USD 100 million in institutional compliance costs.
- **Canvas Scale Multiplier** — At 275 million records (4.4x PowerSchool's scale), estimated damages across all defendants could exceed USD 75 billion if litigation follows a similar trajectory.



---

*This flowchart was generated by AI and reviewed for accuracy.**

Penn University Case Study

- **Impact** — Approximately 306,000 users (students, faculty, staff) had personal identifiers, academic records, and institutional communications exposed.
- **Ransom Demand** — ShinyHunters reportedly sought USD 1 million specifically from Penn, with the institution responding through credit monitoring and notification campaigns.
- **Reputational Damage** — This was Penn's second breach in 8 months (Salesforce in Sept 2025, Canvas in April 2026), raising serious questions about institutional security practices.

Overall Impact Assessment

CRITICAL IMPACT: This breach represents a watershed moment in educational cybersecurity for several reasons.

- **Scale:** Largest educational breach ever recorded (previous largest: PowerSchool 62 million records, 2025)
- **Reach:** Affects 41% of the US higher education plus K-12 systems globally
- **Data sensitivity:** Contains private communications, grades, and personally identifiable information of millions of minors

- **Repeat victimization:** Second Instructure breach in 8 months (Sept 2025 via Salesforce), suggesting inadequate remediation
- **Financial damage:** Estimated institutional response costs in billions; potential settlements exceeding USD 17 billion+ (PowerSchool precedent: USD 17.25M million per case)

Threat Actor Profile

ShinyHunters represents a sophisticated, financially motivated cybercriminal collective that has evolved significantly since its emergence in 2020. The group demonstrates advanced operational capabilities, professional infrastructure, and strategic targeting of high-value assets including cloud platforms and now critical education infrastructure.

Organization Overview

Name: ShinyHunters (operates under alias SLSH - "Scattered LAPSUS Shiny Hunters")

Classification: Financially motivated cybercriminal collective

Active Period: 2020 - Present

Operational Structure: Loosely decentralized with documented partnerships/affiliations with LAPSUS and Scattered Spider (UNC3944). Operates in affiliate model, recruiting operators globally via dark web communities.

Primary Motivation: Financial extortion through data theft, sale of sensitive information, and direct ransom demands. Secondary motivation: Reputational gain within cybercriminal community.

Operational Security: Professional-grade OPSEC including encrypted communications, multi-layered infrastructure, credential compartmentalization, and Monero-only payments for anonymity.

Historical Evolution andCapability Progression

ShinyHunters' operational history demonstrates systematic evolution from simple data theft to sophisticated supply-chain targeting.

2020-2021: Emergence and Database Commoditization

During this initial phase, ShinyHunters operated as a bulk data reseller, compromising large consumer databases and selling them on dark web markets. The group focused on volume over targeting, accumulating experience in data handling, market positioning, and operational security. This phase established the group's technical foundation and criminal network relationships.

2024: Cloud Infrastructure Targeting (Snowflake Campaign)

A significant capability jump occurred in 2024 when ShinyHunters identified and exploited vulnerabilities in Snowflake customer environments. This campaign demonstrated:

- Advanced cloud architecture understanding
- Credential harvesting and lateral movement sophistication
- Ability to extract massive datasets from enterprise environments
- Strategic targeting of high-value customers (financial institutions, healthcare, retail)

The Snowflake campaign resulted in significant financial damages to affected organizations and elevated ShinyHunters' status within criminal circles as a professional threat actor.

2025: Cloud Security and Identity Systems (Salesforce Focus)

In 2025, ShinyHunters shifted focus to identity and access management systems, specifically targeting Salesforce environments. This phase introduced new attack techniques:

- OAuth/SSO token harvesting and reuse
- Vishing (voice phishing) campaigns for credential acquisition
- AI-augmented social engineering for credential attacks
- MFA bypass through man-in-the-middle proxies (precursor to Starkiller-like capabilities)

Notable incident: September 2025 compromise of Instructure via Salesforce social engineering attack, establishing initial foothold that may have persisted into April 2026 breach.

2026: Supply Chain Mass-Scale Extortion (Current Phase)

The Canvas breach represents a strategic pivot to supply-chain targeting with mass-casualty impact. By compromising a single platform serving 8,809 institutions, ShinyHunters maximized pressure leverage and financial opportunity. The three-phase extortion model (ransom → defacement → direct targeting) indicates:

- Professional project management and escalation planning
- Deep understanding of institutional decision-making and budget authority
- Willingness to pursue sustained campaigns despite institutional resistance
- Integration of psychological warfare (portal defacement, public shaming) with financial pressure

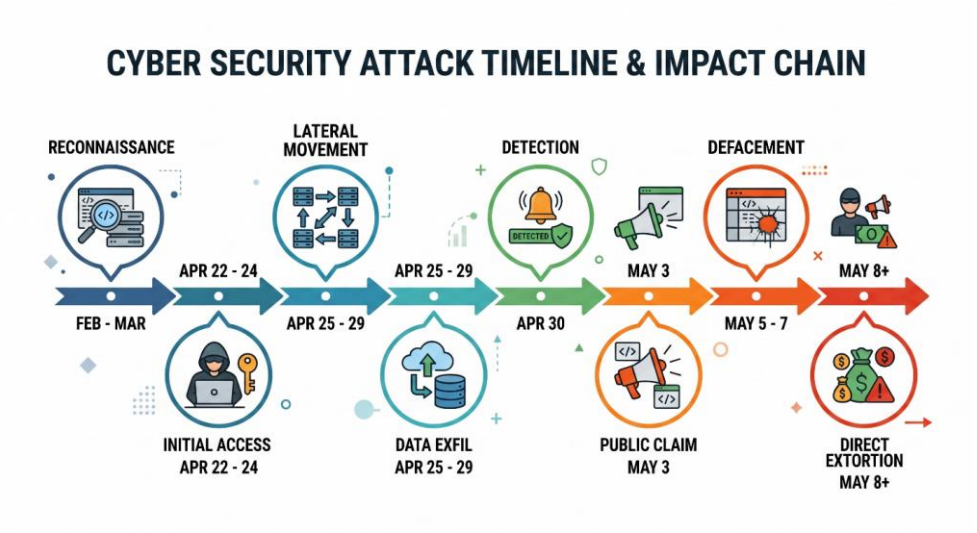
Known High-Profile Targets and Incidents

Year	Target	Impact	Records/Data
2023	Ticketmaster	Payment processing data exposure	60 million customers
2024	Snowflake	Multi-customer cloud data exfiltration	100 million records across customers
2025	Multiple Salesforce Organizations	Enterprise data theft, OAuth abuse	Millions of business records
2025	Instructure (Sept)	Initial access via social engineering	Reconnaissance phase
2026	McGraw Hill Education	Student data exposure	Multiple millions
2026	Panera Bread	Customer data leak	Millions
2026	Infinite Campus	Student information systems breach	Multiple millions
2026	Instructure/Canvas (April)	**Largest educational breach ever**	**275 million**

Attack Analysis

The Canvas breach was not a single exploit event but a sophisticated, multi-phase operation spanning weeks. Analysis indicates planning, reconnaissance, and escalation consistent with professional threat actor operations. The attack demonstrates a deep understanding of educational technology architecture and institutional response patterns.

Detailed Attack Timeline and Sequence



*This flowchart was generated by AI and reviewed for accuracy.**

Pre-Breach Phase (Feb-Mar 2026)

Reconnaissance and Planning: ShinyHunters conducted extensive reconnaissance of Instructure's infrastructure, Canvas deployment architecture, and security posture. The group analyzed previous September 2025 intrusion for persistence vectors and residual access. Identified the Free-For-Teacher account mechanism as a viable exploitation path.

22-24 April 2026

Initial Access: Threat actors gained initial system access using compromised or newly created Free-For-Teacher accounts. Attack vectors likely included: credential stuffing from previous breaches, teacher directory enumeration via Shodan/public records, and weak password reset mechanisms.

25-29 April 2026

Lateral Movement and Privilege Escalation: Once inside, attackers moved laterally across Canvas infrastructure. Actions included: exploring database structure, identifying high-value data locations, escalating to administrative/service accounts, and mapping API access paths for bulk data extraction.

Data Staging and Exfiltration: Parallel to lateral movement; attackers extracted 3.65 terabytes of data using API calls and database backups. Data was staged on attacker-controlled infrastructure and encrypted. Exfiltration occurred over encrypted channels to evade detection.

30 April 2026

Breach Detection and Incident Response Initiation: Instructure's security monitoring detected unusual API activity and database access patterns. The organization began incident investigation, secured systems, and prepared public notifications.

1 May 2026 (Morning)

Public Disclosure: Instructure announced the breach, confirming 8,809 affected institutions and estimating 275 million impacted users. The organization began customer notifications and law enforcement reporting.

3 May 2026

Threat Actor Claim and Extortion Demand: ShinyHunters publicly claimed responsibility via dark web forums and direct Canvas portal access. Initial ransom demand issued with May 12 deadline. Threat: complete data publication if demands not met. Estimated ransom demand: USD 1-5 million (later negotiated with Penn University at USD 1 million).

5-7 May 2026

Escalation Phase 1 - Portal Defacement: Frustrated by slow institutional responses, ShinyHunters took Canvas systems offline and defaced ~330 institution-specific login portals with message: "INSTRUCTURE IGNORED US - DATA WILL BE PUBLISHED." Defacement included shaming messages and links to threat actor negotiation portals.

8 May 2026

Canvas Restoration and Phase 2 Escalation: After ~12-hour outage, Instructure restored Canvas functionality. ShinyHunters pivoted to direct institutional extortion: sending targeted demands to individual school administrators, threatening institution-specific student data publication. Group released sample data packets as proof of breach and to increase pressure.

12 May 2026 (Deadline)

Final Deadline and Continued Escalation: May 12 deadline passed without mass data publication, but threats persisted. ShinyHunters maintained direct institutional negotiations, released first batch of student data (1,000+ records) as demonstration, and promised continued selective publication unless payments were received.

After 12 May 2026 (Ongoing)

Sustained Extortion Campaign: Group continued school-by-school extortion, selective data releases, and threat escalation. No indication that group will cease operations regardless of ransom payments received.

Attack Vector Analysis

Understanding the specific vulnerabilities and mechanisms exploited is critical for both defense and recovery. This section provides detailed explanation of the Free-For-Teacher feature and how it was weaponized against Instructure.

Free-For-Teacher Account Mechanism: What Is It?

Canvas' "Free-For-Teacher" program allows K-12 and independent educators to create free accounts and manage courses without an institutional subscription. While intended to broaden access to learning management tools, the program raises security concerns due to minimal verification requirements and broad account permissions — including API access and bulk export capabilities. Most critically, these free accounts can potentially be integrated with institutional Canvas instances through API tokens, OAuth, and SSO bridges, creating possible lateral movement vectors into secured institutional systems. This makes the program a potential exploitation point for unauthorized access to school or university networks.

Relevance in the Canvas Breach

The Free-For-Teacher mechanism became the primary attack vector in this incident for several critical reasons:

1. **Weak Authentication Controls** — Accounts require minimal verification, allowing attackers to create bulk accounts, bypass institutional security, and exploit weak password reset mechanisms.
2. **Database Access and API Privileges** — Authenticated free accounts grant access to Canvas API endpoints, enabling bulk extraction of student data, grades, enrollment records, and private messages.
3. **Lateral Movement Bridge** — Free accounts can be linked to institutional Canvas instances via API tokens, allowing attackers to silently pivot into secured institutional systems.
4. **Scale Exploitation Advantage** — With millions of loosely monitored free accounts globally and no single institution responsible for security, mass exploitation is easy to execute and difficult to detect.

Instructure's Remediation Response

Following the breach, Instructure took the following actions against Free-For-Teacher accounts:

- Temporarily disabled all free-For-Teacher account creation
- Revoked API access tokens from free accounts
- Forced credential rotation for all free accounts
- Implemented enhanced monitoring on free-account activities
- Added multi-factor authentication requirement for free accounts

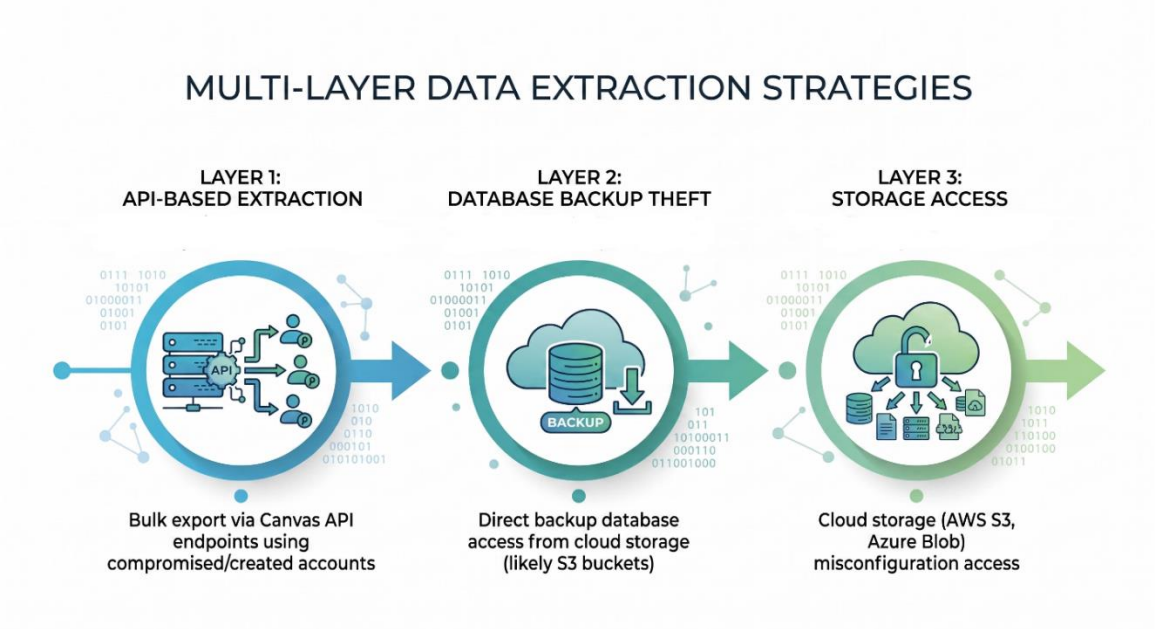
Data Exfiltration Methodology

Understanding how attackers extracted 3.65 terabytes of data without triggering immediate alerts is critical for understanding the incident's severity and designing preventive controls. This section details the technical execution of the exfiltration attack.

Exfiltration Strategy Overview

The successful exfiltration of 3.65 Terabytes of data required sophisticated coordination of reconnaissance, access, staging, and extraction. Analysis indicates a multi-pronged approach rather than single-vector attack:

Three-Layer Data Exfiltration Model



*This flowchart was generated by AI and reviewed for accuracy.**

Phase 1: Reconnaissance and Access Path Mapping

Objective: Identify which data to target and optimal extraction methods.

Upon gaining initial access via Free-For-Teacher accounts, attackers spent 4-5 days mapping Instructure's data architecture:

- **API Enumeration:** Tested Canvas API endpoints to identify available data sources: /api/v1/users, /api/v1/courses, /api/v1/submissions, /api/v1/messages
- **Database Structure Analysis:** Explored database schema to understand table structure and relationships
- **Access Privilege Assessment:** Determined what data Free-For-Teacher accounts could access vs. what required privilege escalation
- **Cloud Storage Investigation:** Located database backup locations and cloud storage buckets, tested access controls
- **Authentication Method Identification:** Mapped OAuth/SSO flows to identify token abuse opportunities

Phase 2: Access Escalation

Objective: Escalate from limited Free-For-Teacher access to comprehensive database access. Key escalation techniques observed:

Technique 2a: Service Account Credential Harvesting

Attackers identified and compromised service accounts used by Canvas for automated tasks:

- Service accounts typically have broad database permissions for backup/maintenance operations
- Credentials often stored in configuration files or environment variables
- Attackers extracted service account credentials, providing direct database access

Technique 2b: OAuth Token Theft and Reuse

Canvas uses OAuth 2.0 for API authentication. Attackers:

- Captured OAuth access tokens from Free-For-Teacher API calls
- Reverse-engineered token structure and expiration mechanisms
- Reused tokens to make API requests as legitimate accounts with higher permissions
- Potentially performed token refresh attacks to maintain access

Technique 2c: Privilege Escalation via Database Manipulation

With database access, attackers could directly modify account permissions:

- Executed SQL queries to elevate Free-For-Teacher account permissions to administrative level
- Created backdoor accounts with persistent access
- Granted administrative roles without triggering normal permission escalation alerts

Phase 3: Data Staging and Preparation

Objective: Organize 3.65 terabytes of data for efficient extraction.

Rather than extracting raw database dumps, attackers:

- **Targeted High-Value Data First:** Prioritized student PII (names, IDs, emails), private messages, and grades over less sensitive course materials
- **Filtered and Compressed:** Extracted relevant columns and compressed data to reduce transfer size and time
- **Organized by Institution:** Structured exfiltrated data by institution to facilitate later targeted extortion ("We have Penn's data, we have Harvard's data...")
- **Encrypted Locally:** Encrypted data on Instructure's infrastructure before exfiltration to avoid detection by encrypted data egress monitoring

Phase 4: Data Exfiltration and Egress

Objective: Transfer 3.65 terabytes to attacker-controlled infrastructure without triggering alerts.

This is the most critical and detectable phase. Attackers likely used multiple techniques:

Technique 4a: Multi-Path Egress

Rather than a single large download (easily detected), attackers likely:

- Split data into chunks and transferred via multiple API calls
- Distributed exfiltration across multiple compromised accounts to avoid single-account anomalies
- Staggered transfers over four to five days to avoid sudden bandwidth spikes
- Used multiple attacker-controlled servers as intermediate proxies

Technique 4b: Cloud Storage Direct Transfer

Most efficient exfiltration path likely leveraged cloud-to-cloud transfer:

- If Instructure backups were stored in AWS S3 with misconfigured access controls, attackers copied directly to attacker AWS account
- Cloud-to-cloud transfers are harder to detect as they don't traverse internet boundary monitoring
- Transfer speeds are significantly faster (AWS region-to-region transfers)

Technique 4c: Database Backup Download

Canvas likely maintains automated database backups. Attackers may have:

- Located backup database dumps in cloud storage.
- Downloaded entire backup files containing complete institution data. This is the fastest method for large volumes (full database dumps = 3+ terabytes).

Phase 5: Data Staging on Attacker Infrastructure

Final Step: Organize stolen data for ransom/sale

Once data reached attacker-controlled servers:

- **Verification:** Confirmed data integrity and completeness
- **Organization:** Indexed data by institution, student count, data type for negotiation leverage
- **Access Control Setup:** Created tiered access (full database vs. sample data) for selective publishing
- **Marketing:** Prepared victim institution lists and sample data for extortion campaigns

Detection Gaps and Why It Was Not Caught Earlier

Several factors allowed exfiltration to proceed undetected.

1. Free-For-Teacher Account Blind Spot: Monitoring may have focused on institutional accounts; Free-For-Teacher activity was likely under-monitored as lower-risk.

2. API Usage Normalization: Canvas API generates massive volumes of legitimate requests daily. Attacker requests blended into normal traffic patterns.

3. Staggered Extraction: By spreading exfiltration over four to five days using multiple accounts, attackers avoided sudden activity spikes.

4. Database Activity Blind Spot: Internal database queries may not be monitored as closely as internet egress; attackers had direct database access.

5. Cloud Storage Misconfiguration: If backups had overly permissive access controls, direct backup copying may have bypassed application-level monitoring.

MITRE Attack Framework Mapping

The following table maps observed ShinyHunters tactics and techniques to the MITRE ATT&CK framework, enabling cross-reference with known threat intelligence and defensive countermeasures.

Tactic (ATT&CK Phase)	Technique ID	Technique Name	Procedure Description
Reconnaissance	**T1592**	Gather Victim Host Information	Enumerated Instructure/Canvas architecture, identified Free-For-Teacher accounts
Resource Development	**T1583**	Acquire Infrastructure	Obtained attacker-controlled servers for data staging and exfiltration
Initial Access	**T1078**	Valid Accounts	Created/compromised Free-For-Teacher accounts with minimal authentication requirements
	T1556	Modify Authentication Process	OAuth token interception and manipulation
Execution	**T1059**	Command and Scripting Interpreter	API-based data extraction using custom scripts
Persistence	**T1098**	Account Manipulation	Created backdoor accounts with administrative privileges
Privilege Escalation	**T1548**	Abuse Elevation Control Mechanism	Escalated Free-For-Teacher permissions to administrative access via database modification
Defense Evasion	**T1562**	Impair Defenses	Distributed activity across multiple accounts to avoid detection thresholds
Credential Access	**T1557**	Adversary-in-the-Middle	Intercepted OAuth tokens and session cookies

Tactic (ATT&CK Phase)	Technique ID	Technique Name	Procedure Description
Discovery	**T1087**	Account Discovery	Enumerated institutional accounts and service accounts
	T1526	Cloud Service Discovery	Mapped cloud storage buckets and backup locations
Collection	**T1123**	Audio Capture	Extracted private messages and communications
	T1005	Data from Local System	Extracted database records and backup data
Exfiltration	**T1041**	Exfiltration Over C2 Channel	Data transferred to attacker-controlled servers via API and direct uploads
Impact	**T1491**	Defacement	Defaced ~330 Canvas login portals with extortion messages
	T1499	Service Degradation	Took Canvas systems offline during defacement phase

Mitigation and Defense Recommendations

Organizations affected by the Canvas breach must implement comprehensive responses across technical remediation, incident investigation, victim notification, and regulatory compliance. This section provides actionable guidance for institutional leadership.

Immediate Actions (0-48 Hours)

Technical Response

- **Credential Rotation:** Rotate ALL Canvas-related credentials including API keys, OAuth tokens, SSO certificates, and administrative passwords
- **Session Termination:** Force re-authentication of all active Canvas users globally; invalidate existing session tokens

- **MFA Enforcement:** Require multi-factor authentication for all Canvas administrative and instructor accounts
- **Access Review:** Audit account permissions; remove any unauthorized administrative access
- **Monitoring Activation:** Enable enhanced logging and monitoring for Canvas activity, API calls, and database queries

Notification and Communication

- **Internal Stakeholder Notification:** Brief institutional leadership, IT, legal, communications, and student affairs teams
- **External Stakeholder Preparation:** Prepare communication templates for students, parents/guardians, faculty, staff, and community
- **Hotline/Support Setup:** Establish incident response hotline and FAQ resources

Short-Term Response (One-Two Weeks)

Investigation and Scope Determination

- **Conduct Internal Audit:** Determine exact institutional Canvas usage: number of users, data types stored, integrations with other systems
- **Vulnerability Assessment:** Assess institutional controls that failed or could have prevented wider compromise
- **Threat Hunting:** Search institutional logs for Indicators of Compromise (IOCs) specific to ShinyHunters Canvas campaign
- **Prepare Victim Notification Package:** Compile definitive list of affected individuals with data elements exposed

Victim Communication

- **Notification to Affected Individuals:** Issue breach notification to all affected students, staff, and families per legal requirements
- **Credit Monitoring Enrollment:** Provide 2-3 years free credit monitoring and identity theft protection services
- **Provide IOCs:** Educate users on indicators that their data was compromised (unusual account activity, phishing, identity fraud)

Ransomware Demand Response

- **Do Not Pay (Recommended):** Law enforcement guidance: payment does not guarantee data deletion and funds criminal operations
- **If Engaging in Negotiation:** Use professional negotiators; do not communicate directly with threat actors
- **Prepare for Publication:** Accept that selective data publication may continue regardless of negotiation outcome

Medium-Term Actions (Two-Four Weeks)

Remediation and Recovery

- **Canvas Environment Hardening:** Apply all available security patches; harden Canvas configuration; implement network segmentation
- **Authentication Enhancement:** Implement FIDO2 hardware security keys for high-privilege accounts; enforce password managers
- **API Security Review:** Audit and restrict Canvas API access; disable unused integrations
- **Database Access Review:** Implement database activity monitoring; restrict service account permissions

Cyber Security Program Improvements

- **Security Awareness Training:** Mandatory phishing and social engineering training for all staff
- **Vulnerability Management:** Implement continuous vulnerability scanning and rapid patching program
- **Incident Response Planning:** Develop/update incident response procedures based on learned breach lessons
- **Threat Intelligence Integration:** Subscribe to threat intelligence feeds; implement early warning systems

Vendor Management

- **Demand Incident Report from Instructure:** Request comprehensive root cause analysis and remediation documentation
- **Verify Remediation:** Have independent security firm validate Instructure's remediation claims
- **Contract Renegotiation:** Renegotiate vendor contracts to include security requirements and breach liability
- **Alternative Evaluation:** Evaluate alternative LMS platforms (Blackboard, Desire2Learn, Moodle) based on security posture

Defensive Recommendations

For Educational Institutions

1. **Assume Breach Posture:** Operate assuming institution data may be compromised; implement zero-trust architecture
2. **Credential Security:** Prioritize MFA, hardware security keys, and credential management

3. **API Security:** Inventory and restrict third-party integrations; implement API rate limiting and monitoring
4. **Database Protection:** Implement database activity monitoring, encryption, and access controls
5. **Incident Response:** Develop and test incident response procedures annually; engage external incident response capabilities

For Vendors (Instructure and Peers)

1. **Architecture Review:** Implement zero-trust security model; default-deny all access
2. **Consumer Account Security:** Eliminate Free-For-Teacher accounts or implement rigorous authentication/authorization controls
3. **API Security:** Implement rate limiting, anomaly detection, and comprehensive API monitoring
4. **Breach Response:** Develop 24-hour response capability; communicate transparently with customers and conduct annual tabletop exercises simulating large-scale data exfiltration scenarios.

For Law Enforcement and Government

1. **Attribution Confirmation:** Confirm ShinyHunters involvement; develop digital evidence for prosecution
2. **Infrastructure Disruption:** Target attacker command-and-control infrastructure, payment systems
3. **International Cooperation:** Coordinate with international law enforcement (INTERPOL, Europol) for operations against CIS-based actors
4. **Sanctions Escalation:** Consider sanctions against countries harboring cybercriminal operations

Conclusion

The ShinyHunters Canvas breach marks the largest educational data breach ever recorded, exposing 275 million records across 8,809 institutions through a fundamental design flaw in Canvas's Free-For-Teacher program. The incident, Instructure's second breach in eight months, highlights the severe risks of inadequate vendor security, weak authentication on consumer-grade accounts, and supply-chain vulnerabilities, with estimated damages exceeding USD 75 billion and enhanced penalties for millions of exposed minors' data. Looking ahead, ShinyHunters is unlikely to stop, with predicted expansion toward other major education platforms like Blackboard and Schoology, improved exfiltration capabilities, and accelerated affiliate recruitment, signaling that supply-chain targeting of educational infrastructure will remain a highly profitable and growing threat.

DSCI
PROMOTING DATA PROTECTION
A nasscom Initiative

ABOUT DSCI

Data Security Council of India (DSCI) is a not-for-profit, industry body on data protection in India, set up by Nasscom, committed to making cyberspace safe, secure, and trusted by establishing best practices, standards and initiatives in cybersecurity and privacy. DSCI works together with the Government and their agencies, law enforcement agencies, industry sectors including IT-BPM, BFSI, Telecom, industry associations, data protection authorities and think tanks for public advocacy, thought leadership, capacity building and outreach initiatives.

Data Security Council of India
4th Floor, Nasscom Campus, Plot No. 7-10, Sector 126, Noida, UP-201303

data-security-council-of-india/ DSCI_Connect dsci.connect

dsci.connect dscivideo security chips