ON2IT AND AMS-IX PRESENT: **THREAT TALKS**

Threat Talks

Supply Chain 2.0

The weakest link breaks the chain

threat-talks.com

The supply chain has become attackers' favorite hunting ground. Every dependency, every third-party update, every open-source component is another potential entry point. And unfortunately, supply chain attacks aren't slowing down.

Instead of kicking down your digital front door, attackers now compromise your partners or software providers and walk in like they belonged here the whole time. It's the modern version of a thief using a key copied from your neighbor's house to simply open yours.

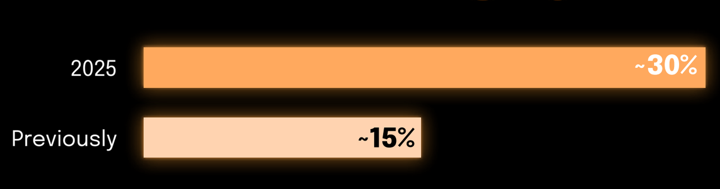
This kind of infiltration turns trust into risk. Even the most secure companies can be exposed through someone else's weakness.

The question isn't whether your defense are strong enough. It's whether the people, tools, and code you rely on are too.

In this Threat Talks infographic we will discuss the following threats:

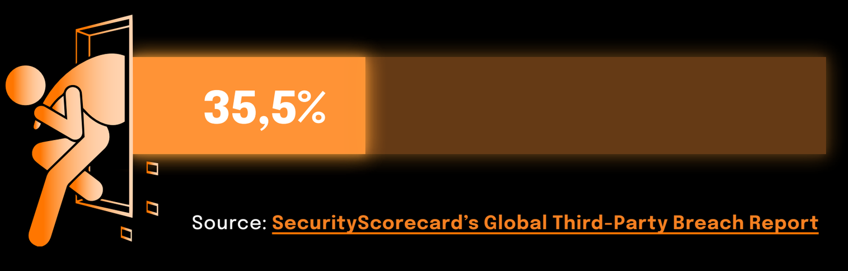
- Crypto Drainer
- Shai-Hulud Worm
- Salesforce

In 2025, the shares of breaches involving a third-party vendor moved to 30% up from ~15% previously



Source: DeepStrike's Supply Chain Statistics 2025

35,5% of breaches in 2024 were linked to third-party access



Source: SecurityScorecard's Global Third-Party Breach Report

47% of organizations suffered a vendor- or supply-chain related disruptive outage in 2024

Source: DIGIT News

The average global cost of a data breach was $4.44 million but supply chain cases cost more + last longer

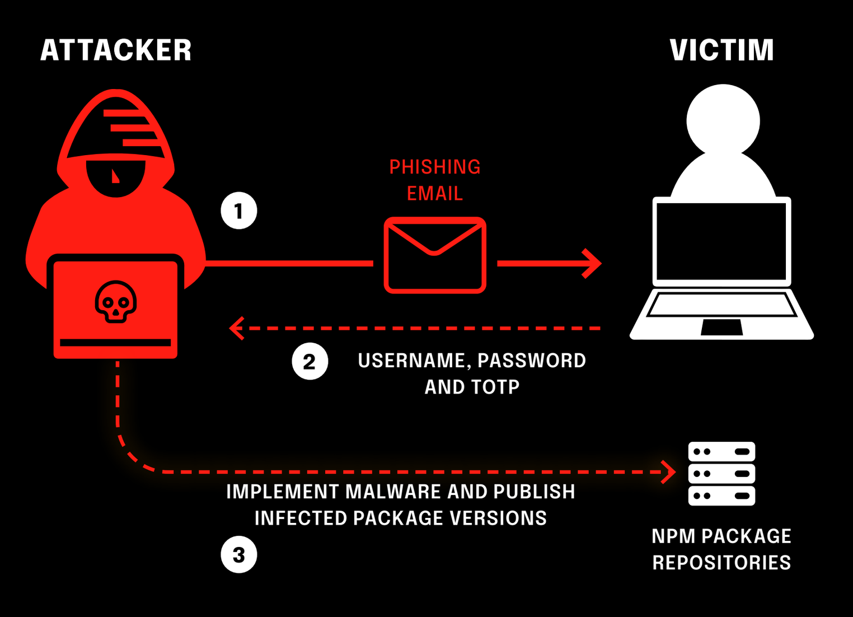
Source: IBM Cost of a Data Breach Report 2025

Crypto Drainer

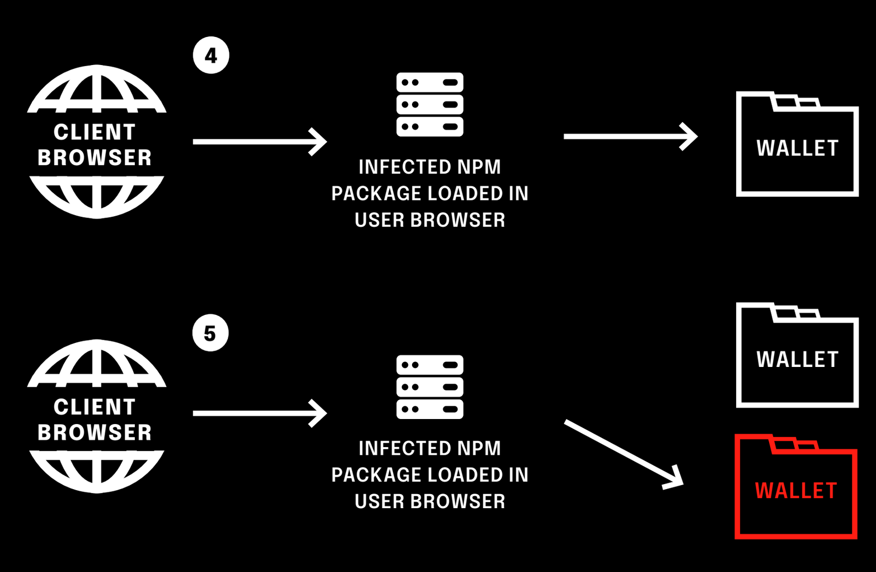
Account Takeover Used to Push Malicious NPM Updates

A phishing email mimicking a 2FA reset tricked a popular NPM maintainer into giving up their username, password and an active TOTP code, enabling full account takeover. The attacker used this access to publish malicious versions of widely used packages like debug and chalk; a set of packages that together see roughly 2 billion downloads per week.. The injected payload was a single line of heavily obfuscated code designed to watch for browser interactions with cryptocurrency wallets and silently redirect transactions to attacker-controlled addresses. Despite the staggering potential blast radius created by the ecosystem's reliance on these utilities, the attack remained narrowly focused on crypto draining and did not achieve widespread real-world impact before the malicious versions were pulled.

INITIAL COMPROMISE



MALWARE WORKINGS



Initial Compromise

1. Attacker sends a phishing mail to the victim.
2. Via the phishing mail, the attacker compromises the victim's username, password and TOTP.
3. The attacker can now implement malware and publish infected package versions.

Malware Workings

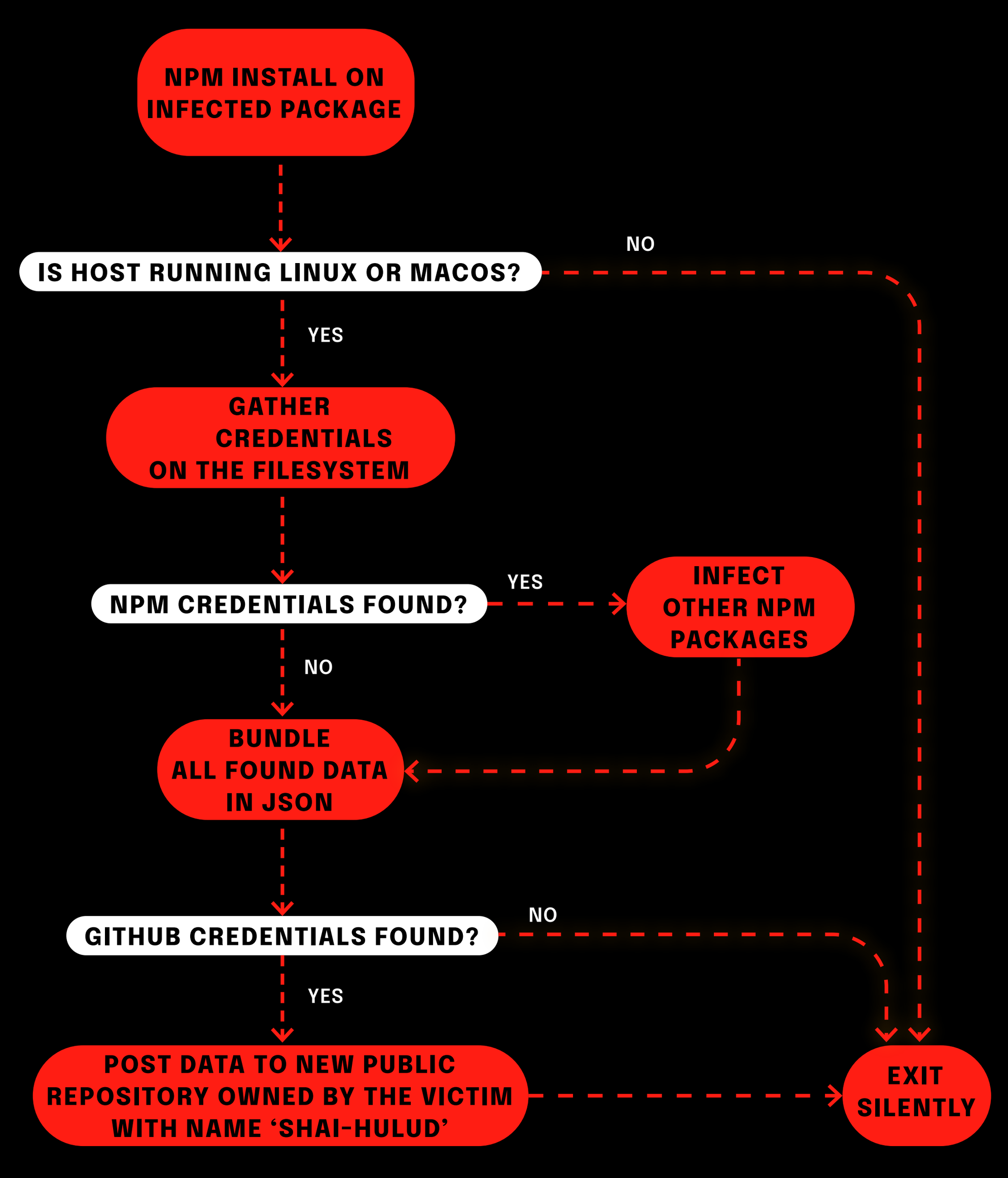
4. The package now watches for calls to wallets related to transactions to wallet applications (like metamask).
5. Once it's spots such a call, the package redirects the transactions to attacked owned addresses / crypte wallets.

Shai-Hulud

Self-spreading and Credential Harvesting Worm

Shai-Hulud was first noticed in the compromised @ctrl/tinycolor package, though its initial entry point is still unknown. Once running, it used TruffleHog to harvest a wide range of credentials, including GitHub tokens, cloud provider keys and NPM auth tokens. It then set up persistence by adding a GitHub Actions workflow that exfiltrated secrets on every push. The worm published stolen credentials to a new public repository named Shai-Hulud under the victim's account and used captured NPM tokens to infect other packages maintained by the victim, pushing new malicious versions to keep spreading. The worm ultimately spread into roughly 640 packages, a cluster that collectively receives hundreds of millions of monthly downloads, giving the attacker access not only to code but also to developer secrets, CI pipelines and cloud accounts.

HOW DOES SHAI-HULUD WORK

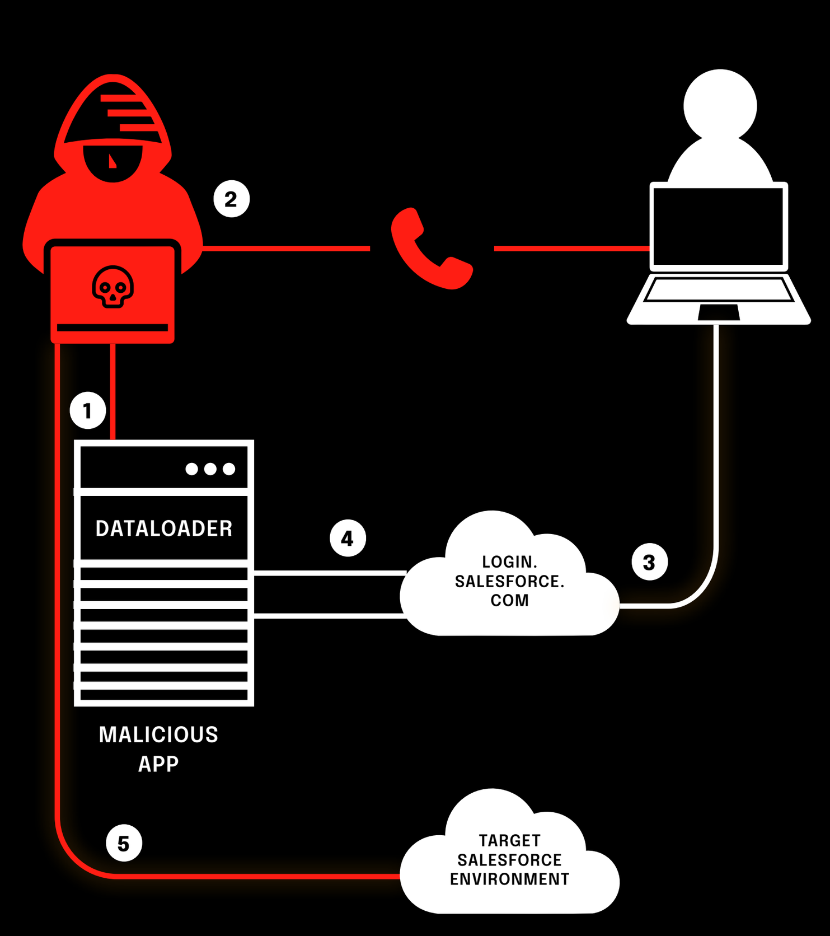


Salesforce

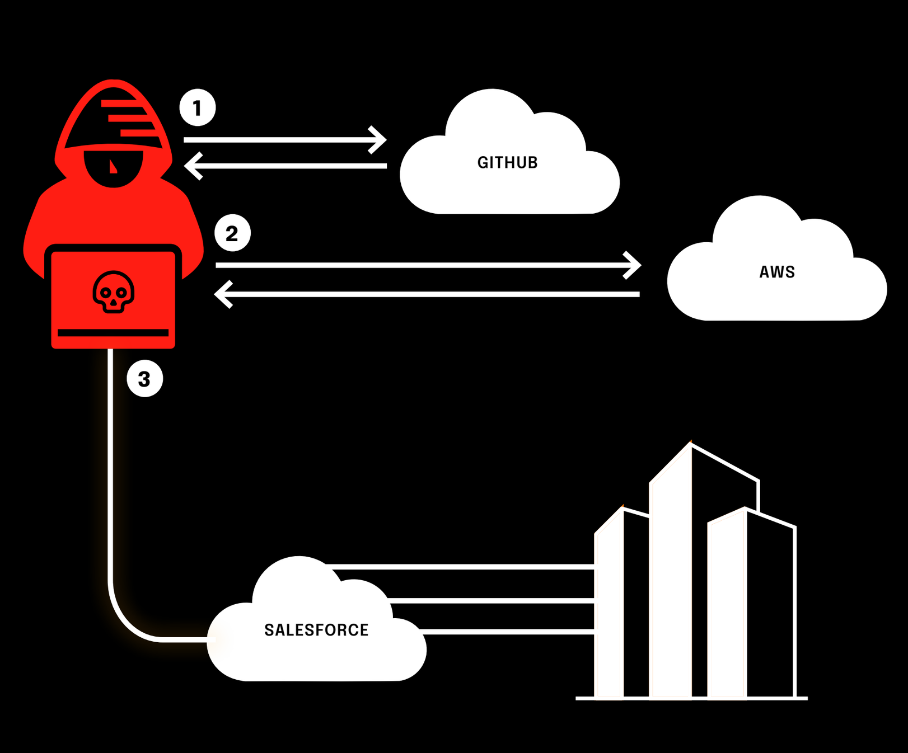
Compromised Integrations Used for Mass Data Access

This attack centered on abusing trusted Salesforce integrations by obtaining OAuth and refresh tokens through upstream compromises. In some cases the attacker convinced users to authorize a malicious connected app; in others they stole tokens directly by breaching third-party vendors and their GitHub or cloud environments. With valid tokens in hand, the attacker gained API-level access to hundreds of customer Salesforce orgs and exfiltrated large volumes of data using SOQL queries and Bulk API jobs. The absence of disclosure creates an unusually opaque blast radius, leaving the scale of compromise – potentially large given the centrality of Salesforce integrations – uncertain.

CAMPAIGN 1



CAMPAIGN 2



Campaign 1

1. **Malicious app**
   Attacker creates a Connected App (often in an org they control), sets an OAuth callback URL they own, and requests broad scopes (e.g., api, refresh_token, sometimes full). Users in the victim org can still authorize that external app via the standard OAuth flow.
2. **Vishing**
   The attacker impersonate IT on a call and steer the user to approve the app, either by visiting the Connected App setup/approval page (/setup/connect) or by clicking an OAuth authorization link the attacker provides.
3. **App connection**
   The user grants consent, the app is now authorized for that user in the victim org under the requested scopes/policies.
4. **Access tokens**
   Salesforce issues an access token (and, if scoped, a refresh token). Subsequent API use presents the bearer token, no new MFA prompt because the user already completed the OAuth consent. With the refresh_token, the attacker can mint new access tokens until you revoke the app/session or change policies.
5. **Data exfiltration**
   Using REST or Bulk API v2, the attacker runs SOQL (e.g., enumerate User, Account, Contact, Case) and bulk-exports records/attachments.

Campaign 2

1. **Initial foothold**
   Attackers gained access to Salesloft's GitHub for months (roughly Mar–Jun 2025), downloaded code from multiple repositories, added a guest user and established workflows used for further access.
2. **Pivot to cloud**
   The attackers accessed Drift's AWS environment and exfiltrated OAuth (and refresh) tokens associated with Drift integrations (incl. the Salesforce chat agent, and later also "Drift Email"). Those tokens let them act as the Drift app against customers' services.
3. **Mass access to customer Salesforce orgs**
   The attackers used the stolen tokens to authenticate to hundreds of Salesforce instances and export large volumes of data using SOQL queries and Bulk API jobs.

Taxonomy

ATT&CK Technique
Which technique of the MITRE ATT&CK framework does the threat correspond to.

ATT&CK Mitigation
Which mitigation of the MITRE ATT&CK framework can be applied.

Attack Strategy
Plan devised by the attacker to exploit specific system vulnerabilities.

Attack Vector
What is the primary method of attack.

Evasion
Tactics used by the attacker to avoid detection or bypass security.

Detection
Mechanism to identify malicious activities or system anomalies.

Complexity
How easy it is to exploit the vulnerability or carry out the attack.

Threat Level
How severe the threat is.

Target Type
The category of organization that may potentially be targeted.

Threat Actor Type
What type of threat actor may be involved.

CONTACT US

mSOC score explanation:
We assign scores to both our sources and the news items. Sources are scored on a numeric scale ranging from 0 (untrustworthy) to 5 (verified), while news items are scored with a letter, ranging from E (unreliable) to A (reliable). By considering the scores of both the source and the news item and the quality of the available information, we classify the overall reliability into three categories: Confirmed, Verified, and Credible. Interested in learning more about our reliability scoring system for sources and news items? Our Threat Intelligence team would be happy to walk you through our procedure, so please don't hesitate to reach out.

threat-talks.com

ZERO TRUST INNOVATORS