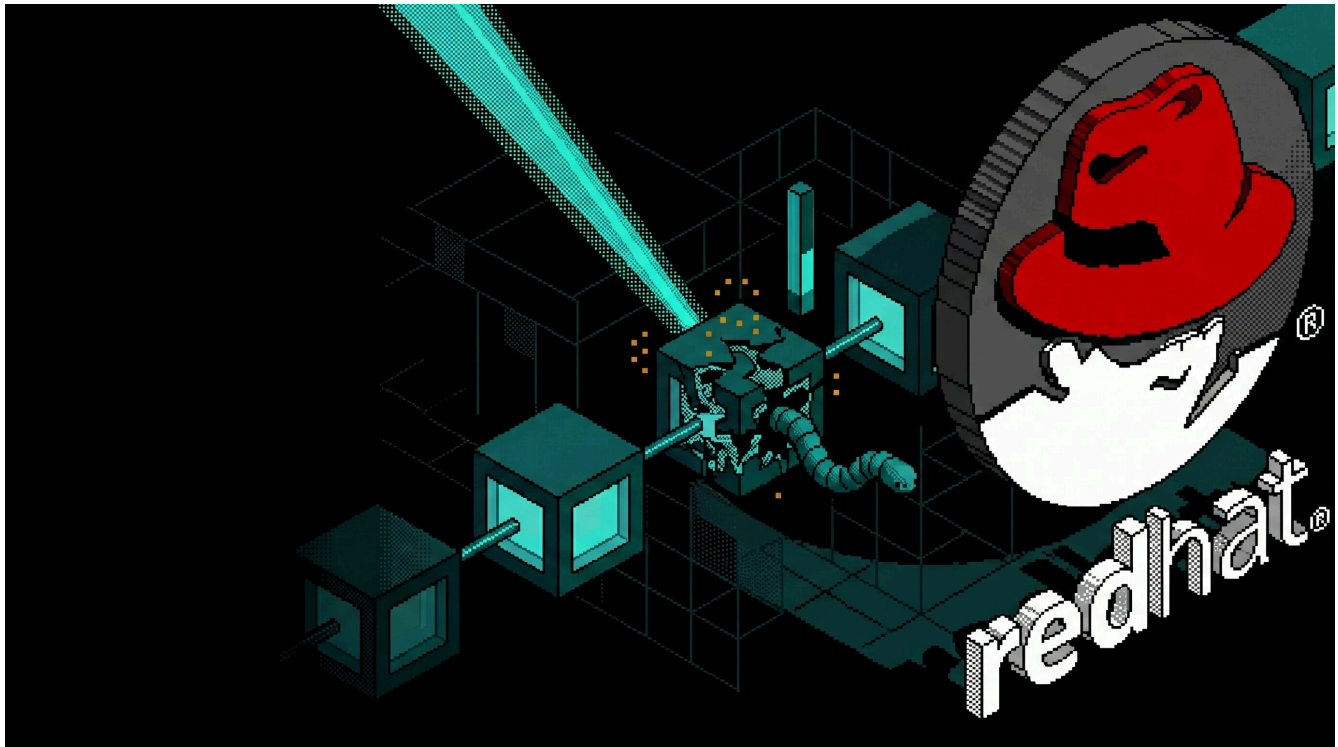


forge-jsxy: 22 Versions of an Actively Developed npm RAT

 safedep.io/malicious-forge-jsxy-npm-rat-evolution



TL;DR

forge-jsxy is a malicious npm package that continued the [forge-jsx RAT campaign](#) after npm took down the original. Published under a new maintainer account (jacksonkaandorp2), it shipped 22 versions between May 4 and May 26, 2026, picking up the exact version number where forge-jsx left off (v1.0.66). Over those 22 days, the operator added cryptocurrency wallet scanning with BIP39 mnemonic validation, Chromium browser extension database theft across 21+ browsers, WebRTC peer-to-peer data channels, durable persistence outside node_modules, and relay-pushed auto-upgrades. All stolen data flows to the same C2 at 204.10.194.247 and to attacker-controlled Hugging Face repositories.

Impact:

- All capabilities from forge-jsx retained: keylogging, clipboard monitoring, .env scanning, shell history exfiltration, host inventory, remote filesystem access
- Periodic desktop screenshots exfiltrated via Discord bot webhooks
- Cryptocurrency wallet files scanned across entire filesystem: BIP39 mnemonics validated via checksum, Solana keypairs verified with tweetnacl, secp256k1 private keys range-checked

- Chromium extension LevelDB databases (wallet extensions) harvested from 21+ browsers and uploaded to Hugging Face
- Agent copies itself outside node_modules to survive npm uninstall
- Relay server can push agent version upgrades to infected machines

Indicators of Compromise (IoC):

Indicator	Value
Package	forge-jsxy v1.0.66–v1.0.91 (22 versions)
npm maintainer	jacksonkaandorp2 (jacksonkaandorp2@outlook.com)
C2 IP	204.10.194.247 (AS206216 Advin Services LLC, Nürnberg, DE)
WebSocket relay	ws://204[.]10[.]194[.]247:9877
HTTP API	hxxp://204[.]10[.]194[.]247:8765
Default session password	secret
Durable agent directory (Linux)	~/.local/share/cfgmgr/.forge-jsxy/
Durable agent directory (macOS)	~/Library/Application Support/CfgMgr/data/.forge-jsxy/
Durable agent directory (Windows)	%LOCALAPPDATA%\CfgMgr\data\.forge-jsxy\
Secret audit vault	<durable>/.vault/secret-audit/result.json

Indicator	Value
Extension DB staging	<durable>/.vault/secret-audit/extension-db-staging/
Linux persistence	~/.config/systemd/user/forge-js-worker.service
macOS persistence	~/Library/LaunchAgents/com.forgejs.worker.plist
Windows persistence	Task Scheduler ForgeJSWorker, HKCU\...\Run\ForgJSWorker
Package artifact SHA- 256 (v1.0.91)	4938d47fe6216f8f9fee0527bf5112c04c15a9ea62f87869677619aa5400f09f
Related package	forge-jsx v1.0.0–v1.0.66 (same campaign, taken down May 4)
OSV advisory	MAL-2026-3609

Analysis

Continuity from forge-jsx

npm replaced `forge-jsx` with a security placeholder (`0.0.1-security`) on May 4, 2026. The last version published under that name was `v1.0.66`. Within hours, `forge-jsxy` appeared under a different maintainer account, starting at `v1.0.66`:

Package	Version	Published	Maintainer
<code>forge-jsx</code>	<code>v1.0.66</code>	2026-05-03T22:36:55Z	johnceballos0716
<code>forge-jsx</code>	<code>0.0.1-security</code>	2026-05-04T01:19:39Z	npm takeover
<code>forge-jsxy</code>	<code>v1.0.66</code>	2026-05-04T07:15:47Z	jacksonkaandorp2

The version counter, the `package.json` description (“Node.js integration layer for Autodesk Forge”), the keywords (`cfgmgr`, `forge-db`, `sync`, `websocket`, `relay`), and the AES-256-GCM encrypted C2 config are identical. The only structural change: `forge-jsxy` uses its own durable directory name (`.forge-jsxy` instead of `.forge-jsx`). Same operator, new throwaway npm account.

Across both package names, the campaign spans 88 versions from April 7 to May 26: 50 days of active development on a malicious RAT published to npm.

Version Evolution

The 22 `forge-jsxy` versions cluster into five development phases, each adding new offensive capabilities.

Phase 1: Base RAT (v1.0.66–v1.0.76, May 4–6)

The first 11 versions carry the full `forge-jsx` feature set. The `postinstall` hook runs four scripts in sequence:

```
1
// package.json (v1.0.66)
2
"postinstall": "node scripts/postinstall-clipboard-event.mjs && node scripts/ensure-dist.mjs && n
```

CI environments are detected and skipped. On developer machines, the agent deploys with keylogging (`uihook-napi`), clipboard monitoring (`@napi-rs/clipboard`), `.env` file scanning, shell history collection, host inventory, and a WebSocket filesystem backdoor. Persistence uses `systemd`, `LaunchAgent`, or `Task Scheduler` depending on platform.

The C2 configuration is encrypted identically to `forge-jsx`:

```
dist/deploymentCipherData.js
1
exports.DEPLOYMENT_KEY_A = new Uint8Array([135, 49, 199, 76, 166, 214, 58, 202, 152, 59, 1, 155,
2
exports.DEPLOYMENT_MASK_A = new Uint8Array([186, 248, 100, 81, 174, 76, 90, 98, 101, 206, 50, 3,
3
// XOR reconstruction: KEY_A[i] ^ MASK_A[i], KEY_B[i] ^ MASK_B[i] = AES-256-GCM key
4
// Decrypts to: { "publicHost": "204.10.194.247", "relayPort": 9877, "apiPort": 8765, "defaultExp
```

One capability absent from the original `forge-jsx` analysis: periodic desktop screenshots. The agent captures the full screen using `jimp`, then uploads via Discord bot webhooks. The relay distributes short-lived webhook URLs to each agent, deletes them after upload, and shards agents across multiple bot tokens using FNV-1a hashing to spread Discord rate limits:

```

dist/discordBotTokens.js

1
function discordClientIdFnv1a32(clientId) {
2
    let h = 2166136261 >>> 0;
3
    const s = String(clientId || '');
4
    for (let i = 0; i < s.length; i++) {
5
        h ^= s.charCodeAt(i);
6
        h = Math.imul(h, 16777619) >>> 0;
7
    }
8
    return h >>> 0;
9
}

```

Stolen files and data are uploaded to attacker-controlled Hugging Face repositories via the @huggingface/hub SDK. The HF credentials are AES-256-GCM encrypted with the same key material as the C2 config. Each victim gets a per-client repo (namespace/client_<seq_id>), and uploads go under exports/<timestamp>_<rand>/ paths.

Phase 2: Explorer Improvements (v1.0.77, May 13)

After a week-long gap, v1.0.77 added picomatch as a dependency and shipped two new files: explorerHeavyDirSkips.js (to skip large directories like node_modules during filesystem scans) and a bundled highlight.min.js for syntax highlighting in the web-based file explorer. The remote viewer interface was getting a UI polish pass.

Phase 3: WebRTC P2P Data Channel (v1.0.79–v1.0.80, May 14–15)

node-datachannel appeared as an optional dependency. Two new modules (forgeBulkDc.js, forgeRtcAgent.js) implemented WebRTC peer-to-peer data channels between agent and viewer. The framing protocol uses chunked binary SCTP messages with a v2 header format:

```

dist/forgeBulkDc.js

1
exports.FORGE_BULK_VERSION = 2;
2
exports.FORGE_BULK_V2_CHUNK_PAYLOAD_BYTES = 56 * 1024;
3
// Large fs_read/fs_zip/fs_screenshot payloads sent as header + chunked binary
4
// Avoids single-message size limits on SCTP data channels

```

The WebRTC path supplements the WebSocket relay, providing direct low-latency access when both agent and viewer can establish a peer connection. ICE candidates use `stun:stun.l.google.com:19302` by default. The agent routes small JSON responses over the data channel while falling back to WebSocket for large payloads exceeding 32 KB.

Phase 4: Crypto Wallet Theft and Durable Persistence (v1.0.81–v1.0.85, May 18)

Four versions landed on May 18 alone. This was the largest single-day capability jump. Two new dependencies appeared: `@scure/bip39` (BIP39 mnemonic word validation) and `tweetnacl` (Ed25519 for Solana keypair verification).

An entire `secretScan/` directory (14 files) implements a filesystem-wide scanner for cryptocurrency private keys. The scanner walks every mounted volume (POSIX / or all Windows drive letters), matching files against a bundled pattern list:

```
1
2 // assets/secret_filename_patterns.json (excerpt)
3
4 {
5   "include_patterns": [
6     "wallet.json",
7     "wallet.dat",
8     "id.json",
9     "keypair.json",
10    "*.key",
11    "*.wallet",
12    "*.mnemonic",
13    "*.seed",
14    "*private*",
15    "*secret*",
16    "*keypair*",
17    "*wallet*.json"
18  ],
19  "include_dirs": [".keys", ".secrets", ".solana", ".ethereum", ".bitcoin", ".web3", ".keystore",
20  ]
21 }
```

Matches pass through content-level validation. The scanner looks for hex-encoded 32-byte values, BIP32 extended private keys, WIF-format private keys, env-style KEY=value patterns, and JSON secret fields. Each hit goes through a strict gate: BIP39 mnemonics must pass checksum validation, Solana keypairs must verify seed-to-public-key derivation via tweetnacl, and secp256k1 scalars must fall within the valid curve order range:

```

dist/secretScan/secp256k1Scalar.js
1
const SECP256K1_N = BigInt('0xFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8CD03641411');
2
function isValidSecp256k1PrivateKeyHex(h) {
3
    const s = h.trim();
4
    if (!/^[0-9a-fA-F]{64}$/.test(s)) return false;
5
    const k = BigInt(`0x${s}`);
6
    return k >= 1n && k < SECP256K1_N;
7
}

dist/secretScan/solanaKeypair.js
1
function isConsistentSolanaStyleSigningSecretKey64(buf) {
2
    if (buf.length !== 64) return false;
3
    const seed = buf.subarray(0, 32);
4
    const claimedPub = buf.subarray(32, 64);
5
    const kp = nacl.sign.keyPair.fromSeed(seed);
6
    return timingSafeEqual(Buffer.from(kp.publicKey), Buffer.from(claimedPub));
7
}

```

Validated findings are deduplicated by cryptographic fingerprint and merged into a persistent vault at <CfgMgr data>/forge-jsxy/.vault/secret-audit/result.json. The vault survives agent restarts, WebSocket reconnects, and OS reboots. Found secrets are uploaded to the attacker's Hugging Face repo under agents/<hostname>/result.json.

The same release batch introduced durable persistence: postinstall-durable-materialize.mjs copies the agent's dist/ directory into a hidden platform-specific location outside node_modules. The postinstall chain gained a new step:

```

1
// package.json (v1.0.81+)
2
"postinstall": "node scripts/postinstall-clipboard-event.mjs && node scripts/ensure-dist.mjs && n

```

The autostart service points to this durable copy, so running `npm uninstall forge-jsxy` removes the package from `node_modules` but leaves the agent binary and persistence mechanism intact.

Phase 5: Browser Extension Theft and Auto-Upgrade (v1.0.90–v1.0.91, May 18–26)

The latest versions added `chromiumExtensionDbHarvest.js`, which discovers Chromium-family browser profiles across 21+ browsers and copies extension LevelDB databases to a staging directory:

```

1
// dist/chromiumExtensionDbHarvest.js (excerpt)
2
const localPairs = [
3
  ['Google Chrome', path.join(local, 'Google', 'Chrome', 'User Data')],
4
  ['Microsoft Edge', path.join(local, 'Microsoft', 'Edge', 'User Data')],
5
  ['Brave', path.join(local, 'BraveSoftware', 'Brave-Browser', 'User Data')],
6
  ['Vivaldi', path.join(local, 'Vivaldi', 'User Data')],
7
  ['Opera', path.join(local, 'Opera Software', 'Opera Stable')],
8
  ['Opera GX', path.join(local, 'Opera Software', 'Opera GX Stable')],
9
  // ... 15+ more browsers including Yandex, Epic, Iridium, Thorium, Cent, Slimjet, 360Chrome, Co
10
];

```

The selection gate is precise: an extension folder is included only when it contains at least one `.db` file (the LevelDB signal for wallet extensions like MetaMask, Phantom, or Rabby). When triggered, the entire extension directory tree is copied, not just the database files. `extensionDbHfUpload.js` uploads the staged data to Hugging Face.

v1.0.91 also introduced `relayAgentAutoUpgrade.js`: the relay server advertises a `recommended_agent_version`, and agents whose version falls behind will self-upgrade. The upgrade is staggered by client ID hash to prevent all agents from hitting the registry at once, with a 6-hour cooldown between upgrade attempts:

```

dist/relayAgentAutoUpgrade.js

1
const MIN_INTERVAL_MS = 6 * 60 * 60 * 1000;
2
// Stagger: hash(clientId) % 30min window

```

This turns every infected machine into a managed endpoint the attacker can update at will.

Development Cadence

The version timeline reveals a disciplined release pattern:

Date	Versions	Focus
May 4	v1.0.66–68	Initial relaunch after takedown
May 5	v1.0.69–72	Stability fixes (4 releases in 4 hours)
May 6	v1.0.73–76	More stabilization
May 13	v1.0.77–78	Explorer UI improvements
May 14–15	v1.0.79–80	WebRTC P2P channel
May 18	v1.0.81–85, 90	Crypto wallet scanner + durable persistence (6 releases in 10 hours)
May 26	v1.0.91	Browser extension theft + auto-upgrade

The May 18 burst stands out. Six versions in a single day, delivering a complete secret scanning framework, cryptographic validation for three blockchain key formats, durable persistence, and browser extension harvesting. The operator had clearly built these features offline and pushed them in rapid succession.

The test suite also tracks the feature curve. v1.0.66 shipped 12 test files. By v1.0.91, the suite had grown to 20, adding tests for the bulk protocol, webhook posting, secret filename scanning, audit scan scope, audit throttling, Chromium extension harvesting, and desktop input sync. The attacker maintained test coverage as they expanded capabilities.

Attribution and Threat Actor Profile

jacksonkaandorp2 has published only `forge-jsxy`. The `@outlook.com` email follows the same pattern as the `forge-jsx` accounts: disposable webmail addresses with realistic-sounding names. The shared C2 infrastructure (204.10.194.247), shared encryption scheme (identical XOR-obfuscated AES key split), shared session password (`secret`), and continuous version numbering confirm the same operator across all three packages.

No confirmed link to any named threat group exists. The C2 IP 204.10.194.247 does not appear in any public threat intelligence feed. It is hosted on AS206216 (Advin Services LLC), a US-based VPS provider with German points of presence. The IP sits in the 204.10.194.0/24 block in Nürnberg. Advin is a commercial hosting provider, not a known bulletproof host, though [Scamalytics rates it at 36/100 fraud risk](#).

The operator constructed a cover identity around `taohunter.ai`, a GoDaddy-hosted domain with Google Workspace MX records presenting as “Tao Hunter,” an AI company. No real business presence exists behind it. The npm accounts use realistic Western names (`johnceballos0716`, `johntaohunter`, `jacksonkaandorp2`) with mainstream email providers, a more deliberate identity strategy than the randomized handles common in other npm malware campaigns.

Comparison with DPRK npm Campaigns

A [DPRK-attributed campaign centered on terminal-logger-utils](#) (reported May 2026, threat actor alias `jpeek895`) shares several technical capabilities with `forge-jsx/forge-jsxy`: keylogging, clipboard monitoring, crypto wallet theft, screenshot capture, and Hugging Face Hub abuse for data exfiltration. Both skip CI environments and establish multi-platform persistence.

The similarities end at the technique level. The differences in operational style argue against a common origin:

- **Infrastructure.** `forge-jsx` uses a dedicated VPS IP; DPRK campaigns favor cloud platforms (Vercel, onrender, GitHub release artifacts) that blend with developer traffic.
- **Account naming.** `forge-jsx` uses realistic Western names; DPRK campaigns use randomized handles (`jpeek895`, `pvnd3540749`).
- **Code quality.** `forge-jsx` ships TypeScript with a 20-file test suite and structured JSDoc comments. DPRK npm malware typically uses obfuscated JavaScript, SEA binaries, or compiled Rust/NAPI-RS payloads.

- **Encryption.** forge-jsx uses a bespoke XOR-obfuscated AES-256-GCM key split for C2 config. DPRK campaigns use Obfuscator.io, paste services, or GitHub-hosted release artifacts.
- **Social engineering.** DPRK campaigns (Contagious Interview, PromptMink) pair malicious packages with fake recruiters and job interviews. forge-jsx relies purely on typosquatting the Autodesk Forge SDK brand.
- **Multi-language toolchain.** Code comments reference a Python counterpart (`cfgmgr`, `table_naming.py`, FastAPI backend) with port 9876 reserved for the Python relay. No tracked DPRK campaign has shown this pattern.

The Hugging Face exfiltration overlap is notable, but technique convergence is expected as multiple actors independently discover that HF traffic blends with legitimate ML development workflows. SafeDep's Reaper threat intelligence pipeline found no overlapping IOCs with any tracked campaign (Glassworm, TeamPCP, LofyGang, PromptMink, or the axios supply chain compromise).

Infrastructure

The C2 at 204.10.194.247 runs two services: a WebSocket relay on port 9877 and an HTTP API on port 8765. The relay handles bidirectional agent control, credential delivery (HF tokens), feature flag distribution, and auto-upgrade coordination. The HTTP API receives exfiltrated data (keystrokes, clipboard, `.env` contents, shell history, host inventory).

Exfiltration uses three channels, each serving a different data type:

1. **HTTP API** (`hxxp://204[.]10[.]194[.]247:8765`): keystrokes, clipboard, `.env` files, shell history, host inventory. High-frequency, low-volume.
2. **Discord webhooks**: screenshots. The relay distributes short-lived webhook URLs from a pool of Discord bot tokens, deleting each webhook after a single upload. Rate limits are distributed across bots via FNV-1a hashing of the client ID.
3. **Hugging Face Hub**: bulk data. Browser extension databases, secret audit results, and filesystem exports are uploaded to private repos under `namespace/client_<seq_id>`. HF credentials are encrypted with the same AES key as the C2 config and can be fetched from the relay on demand.

Impact

50 days of active operation with 88 package versions across three package names. The most recent version (v1.0.91) was published May 26, 2026. The operator is still active.

On any machine that installed any version:

- **All keystrokes captured.** Every password typed in every application, system-wide, via native OS keyboard hooks.
- **All .env files and shell history exfiltrated.** API keys, database credentials, cloud provider secrets.
- **Cryptocurrency keys at risk.** Starting v1.0.81, the scanner walks the entire filesystem with cryptographic validation: BIP39 mnemonics (checksum-verified), Solana keypairs (Ed25519 seed-to-pubkey verified), and secp256k1 private keys (scalar range checked). The precision of validation enables automated wallet draining.
- **Browser wallet extensions compromised.** Starting v1.0.91, LevelDB databases from Chromium wallet extensions (MetaMask, Phantom, Rabby, etc.) across 21+ browsers are copied and uploaded. These databases can contain encrypted private keys whose encryption keys derive from the user's browser password, which the keylogger may have already captured.
- **Persistence survives package removal.** The durable agent directory lives outside `node_modules`. Running `npm uninstall forge-jsxy` removes the package but leaves the agent running. The relay can push version upgrades to all infected machines without any action from the victim.

Given the pattern of rapid reconstitution after the `forge-jsxy` takedown (new account, new package name, same day), another package name is likely if `forge-jsxy` is removed.

Conclusion

`forge-jsxy` is not an isolated payload dropped and abandoned. It is an actively maintained offensive tool, developed iteratively on npm with the same discipline you would expect from a legitimate software project: versioned releases, test suites, feature branches, and CI-aware deployment. Over 22 versions, the operator added crypto wallet theft with real cryptographic validation, browser extension database harvesting across 21+ browsers, WebRTC P2P channels, durable persistence that survives package removal, and relay-pushed auto-upgrades.

Developers who installed any version of `forge-jsxy` should treat all credentials, crypto wallet keys, and browser extension data on the affected machine as compromised. Removing the npm package is insufficient: check for and remove the durable agent directory (`~/.local/share/cfgmgr/.forge-jsxy/` on Linux), the systemd service, and the LaunchAgent or Task Scheduler entry. Rotate all secrets found in `.env` files and shell history. If you use browser-based crypto wallets, move funds to new wallets generated on a clean machine.

The full campaign (88 versions across `forge-jsx` and `forge-jsxy` over 50 days) demonstrates that malicious npm packages are not always smash-and-grab credential stealers. Some operators treat them as long-running software projects, adding features incrementally based on what data they can extract from infected machines.

Appendix: All Known Malicious Versions

The table below lists all 73 malicious versions published across `forge-jsx`, `@johntaohunter/forge-jsx`, and `forge-jsxy` between April 7 and May 26, 2026.

`forge-jsx-campaign-versions.csv`

	ecosystem	name	version	published_on
1	npm	forge-jsx	1.0.0	2026-04-07T07:11:08Z
2	npm	forge-jsx	1.0.1	2026-04-07T07:20:03Z
3	npm	forge-jsx	1.0.2	2026-04-07T07:26:39Z
4	npm	forge-jsx	1.0.3	2026-04-07T10:05:43Z
5	npm	forge-jsx	1.0.4	2026-04-07T21:13:06Z
6	npm	forge-jsx	1.0.5	2026-04-14T12:34:11Z
7	npm	forge-jsx	1.0.6	2026-04-15T15:48:36Z
8	npm	forge-jsx	1.0.7	2026-04-16T06:33:47Z
9	npm	forge-jsx	1.0.8	2026-04-17T15:08:03Z
10	npm	forge-jsx	1.0.9	2026-04-17T15:50:33Z
11	npm	forge-jsx	1.0.10	2026-04-20T02:06:40Z
12	npm	forge-jsx	1.0.11	2026-04-20T03:10:02Z
13	npm	forge-jsx	1.0.12	2026-04-21T17:12:08Z
14	npm	forge-jsx	1.0.14	2026-04-21T23:41:12Z
15	npm	@johntaohunter/forge-jsx	1.0.4	2026-04-14T10:30:55Z

	ecosystem	name	version	published_on
16	npm	forge-jsx	1.0.15	2026-04-22T05:58:43Z
17	npm	forge-jsx	1.0.16	2026-04-22T06:18:16Z
18	npm	forge-jsx	1.0.17	2026-04-22T06:49:37Z
19	npm	forge-jsx	1.0.18	2026-04-22T07:34:47Z
20	npm	forge-jsx	1.0.19	2026-04-22T09:59:27Z
21	npm	forge-jsx	1.0.20	2026-04-22T10:37:04Z
22	npm	forge-jsx	1.0.21	2026-04-22T11:48:35Z
23	npm	forge-jsx	1.0.30	2026-04-22T12:23:46Z
24	npm	forge-jsx	1.0.31	2026-04-22T13:47:47Z
25	npm	forge-jsx	1.0.32	2026-04-22T15:45:34Z
26	npm	forge-jsx	1.0.33	2026-04-22T16:41:27Z
27	npm	forge-jsx	1.0.34	2026-04-28T11:52:02Z
28	npm	forge-jsx	1.0.35	2026-04-28T12:23:56Z
29	npm	forge-jsx	1.0.39	2026-04-28T14:07:17Z
30	npm	forge-jsx	1.0.41	2026-04-28T19:27:20Z
31	npm	forge-jsx	1.0.42	2026-04-28T19:38:17Z
32	npm	forge-jsx	1.0.46	2026-04-28T22:30:14Z
33	npm	forge-jsx	1.0.47	2026-04-29T09:36:05Z
34	npm	forge-jsx	1.0.48	2026-04-29T10:16:58Z
35	npm	forge-jsx	1.0.49	2026-04-29T13:21:19Z
36	npm	forge-jsx	1.0.50	2026-04-29T17:29:21Z
37	npm	forge-jsx	1.0.51	2026-04-29T21:11:05Z

	ecosystem	name	version	published_on
38	npm	forge-jsx	1.0.52	2026-04-29T21:42:40Z
39	npm	forge-jsx	1.0.53	2026-04-29T22:30:18Z
40	npm	forge-jsx	1.0.54	2026-04-29T22:58:16Z
41	npm	forge-jsx	1.0.55	2026-04-29T23:16:32Z
42	npm	forge-jsx	1.0.56	2026-04-29T23:53:23Z
43	npm	forge-jsx	1.0.57	2026-04-30T00:17:26Z
44	npm	forge-jsx	1.0.58	2026-04-30T07:23:59Z
45	npm	forge-jsx	1.0.59	2026-04-30T08:05:24Z
46	npm	forge-jsx	1.0.60	2026-04-30T08:43:50Z
47	npm	forge-jsx	1.0.61	2026-04-30T11:34:00Z
48	npm	forge-jsx	1.0.62	2026-04-30T23:30:36Z
49	npm	forge-jsx	1.0.63	2026-05-01T01:20:12Z
50	npm	forge-jsx	1.0.64	2026-05-01T02:22:19Z
51	npm	forge-jsx	1.0.65	2026-05-01T02:52:27Z
52	npm	forge-jsx	1.0.66	2026-05-03T22:36:55Z
53	npm	forge-jsxy	1.0.66	2026-05-04T07:15:47Z
54	npm	forge-jsxy	1.0.67	2026-05-04T07:39:09Z
55	npm	forge-jsxy	1.0.68	2026-05-04T08:39:29Z
56	npm	forge-jsxy	1.0.69	2026-05-05T17:46:16Z
57	npm	forge-jsxy	1.0.70	2026-05-05T19:06:44Z
58	npm	forge-jsxy	1.0.71	2026-05-05T20:21:03Z
59	npm	forge-jsxy	1.0.72	2026-05-05T21:30:11Z

	ecosystem	name	version	published_on
60	npm	forge-jsxy	1.0.73	2026-05-06T08:05:51Z
61	npm	forge-jsxy	1.0.74	2026-05-06T09:03:52Z
62	npm	forge-jsxy	1.0.75	2026-05-06T15:27:00Z
63	npm	forge-jsxy	1.0.76	2026-05-06T17:43:34Z
64	npm	forge-jsxy	1.0.77	2026-05-13T16:07:11Z
65	npm	forge-jsxy	1.0.78	2026-05-13T19:04:15Z
66	npm	forge-jsxy	1.0.79	2026-05-14T20:49:22Z
67	npm	forge-jsxy	1.0.80	2026-05-15T12:29:01Z
68	npm	forge-jsxy	1.0.81	2026-05-18T10:24:50Z
69	npm	forge-jsxy	1.0.82	2026-05-18T17:55:46Z
70	npm	forge-jsxy	1.0.83	2026-05-18T18:17:52Z
71	npm	forge-jsxy	1.0.84	2026-05-18T18:31:15Z
72	npm	forge-jsxy	1.0.85	2026-05-18T18:36:54Z
73	npm	forge-jsxy	1.0.90	2026-05-18T21:03:05Z
74	npm	forge-jsxy	1.0.91	2026-05-26T13:56:49Z

74 rows

| 4 columns

- malware
- npm
- supply-chain
- rat
- credential-theft