Java Applets in Airline Applications

Risk Assessment & Industry Impact

Supplement to: Browser Risk & Commercial Comparison Report (IE11 · Chrome 44 · Edge)

Mike Irons
Sales@elevationsoftware.com
April 28, 2026

Table of Contents

1. Executive Summary

Java Applets were once the dominant technology for delivering cross-platform, browser-based operational interfaces in the aviation industry — powering DCS check-in terminals, CUTE agent workstations, CUSS kiosks, and ground handling systems at airports worldwide. That era is definitively over.

NPAPI — the browser plugin architecture that enabled Java Applets to run — was killed by Chrome in 2015 and Firefox in 2017. Oracle deprecated the Java Applet API in Java 9 (2017), removed it from Java 11 (2018), and in March 2026 removed the entire java.applet package from Java 26. As of April 2026, there is no supported browser, no supported JDK, and no viable security posture for any application that still depends on Java Applets.

For airlines and airports still operating Java Applet-based systems, this represents a convergence of three simultaneous crises: security, compliance, and commercial licensing.

[CRITICAL] Java Applets are dead at every level: browser support (since 2015–2017), Oracle JDK support (since 2018), and JDK package existence (since March 2026). Any airline application still invoking Java Applets is running on a foundation that no longer exists in any supported technology stack.

2. The Death of Java Applets — Complete Timeline

Date	Event	Impact on Airline Systems
Sep 2013	Google announces NPAPI phase-out over 2014	Airlines given warning: DCS/CUTE Java Applet interfaces need replacement roadmap
May 2014	Chrome 35: NPAPI removed on Linux	Linux-based airport terminals lose Java Applet support
Apr 2015	Chrome 42: NPAPI disabled by default on all platforms	Java Applets silently broken for most Chrome users at airport workstations
Sep 2015	Chrome 45: NPAPI completely removed	Chrome permanently cannot run any Java Applet — DCS/CUTE interfaces broken for all Chrome users
Sep 2015	Chrome 44 released then superseded (6-week window)	Chrome 44 = last version with NPAPI, but only for ~6 weeks
Jan 2016	Oracle announces Java browser plugin end-of-life	Official end of Oracle-supported Java browser plugin; IE11 Java integration enters unsupported territory
Mar 2017	Firefox 52: NPAPI removed (except Flash)	Firefox permanently cannot run Java Applets — only IE11 ActiveX remains
Sep 2017	Java 9 released: Applet API deprecated	Oracle formally signals Java Applets are end-of-life within the language itself
Mar 2018	Java 11 released: Java Plugin, Appletviewer, Java Web Start all removed	No supported JDK runtime includes browser applet capability. IE11 ActiveX Java remains as last holdout.
Jun 2022	IE11 reaches end of life	The last browser with Java ActiveX support loses vendor patches
Sep 2023	JDK 21 LTS: Applet API deprecated for removal (JEP 398)	Final warning — applet code will not compile in future JDK versions
Jan 2023	Oracle changes Java SE licensing to per-employee model	3–10x cost increase for organisations still licensing Java SE for legacy applet runtimes
Mar 2026	Java 26: entire java.applet package removed (JEP 504)	Java Applets cannot be compiled or run on any current supported JDK. Dead at language level.
Apr 2026	No mainstream browser supports NPAPI / Java Applets	Chrome, Edge, Firefox, Safari: zero support. IE11: unsupported, unpatched.

3. Why the Aviation Industry Relied on Java Applets

Java Applets became the default technology for airport operational interfaces in the late 1990s and 2000s for legitimate technical reasons:

- **Cross-platform:** Write once, run anywhere — Java's cross-platform runtime allowed a single application to run on Windows, Linux, and Solaris airport workstations without separate builds
- **Zero-touch deployment:** Browser-delivered deployment — updates could be pushed to hundreds of airport terminals without physical installation visits
- **UI richness:** Rich GUI capability — Java Swing provided richer interfaces than HTML at the time, critical for complex DCS check-in and boarding workflows
- **Device integration:** Serial/hardware device access — Java Applets with elevated permissions could interface with barcode scanners, passport readers, bag tag printers, and boarding card printers
- **Vendor support:** Vendor ecosystem — Amadeus, SITA, Sabre, and IBM all shipped Java Applet-based DCS and CUTE interfaces from the late 1990s onwards

3.1 Airline Applications That Used Java Applets

Application Type	Description	Java Applet Role
DCS — Departure Control System	Manages check-in, seat assignment, baggage acceptance, boarding, load control	Browser-delivered agent interface; real-time passenger data, seat maps, weight & balance
CUTE — Common Use Terminal Equipment	Shared airport infrastructure allowing any airline agent to use any workstation	Java Applet delivered the airline-specific DCS interface on shared CUTE hardware
CUSS — Common Use Self-Service	Shared kiosk infrastructure for passenger self-check-in	Java Applet ran the check-in application, bag tag printing, and seat selection within the kiosk browser shell
CUPPS — Common Use Passenger Processing	IATA standard for shared check-in and gate podium systems	Java Applet provided the UI layer for agent-facing CUPPS workstation applications
Ground Handling Systems	Ramp, fuelling, catering, and turnaround management	Java Applets delivered operational interfaces to ground crew tablets and workstations
FIDS — Flight Information Display	Departure and arrival board management	Some FIDS management consoles used Java Applets for airport operations centre control
Baggage Reconciliation Systems	Bag tracking and reconciliation against passenger manifest	Java Applet interfaces for baggage agents reading scanner data and reconciling loads
Load Control	Aircraft weight, balance, and loading instructions	Complex calculation interfaces delivered as Java Applets to load controllers

4. Java Applet Security Vulnerabilities

4.1 Why Java Applets Were a Security Disaster

[CRITICAL] Java Applets were the single most exploited browser technology of the 2010s. The Java sandbox bypass became so reliable and repeatable that it was a standard module in every major commercial exploit kit (Blackhole, Neutrino, Magnitude, Angler).

The fundamental problem was that Java Applets ran native code inside a security sandbox that was proven repeatedly to be escapable. Once an attacker escaped the Java sandbox, they had full access to the host operating system — with the privileges of the browser process.

4.2 Major CVEs — Java Applet Sandbox Escapes

CVE	Year	Severity	Description
CVE-2010-0840	2010	CRITICAL	JRE sandbox bypass — exploited by Blackhole exploit kit; one of the first mass-exploitation Java CVEs
CVE-2012-0507	2012	CRITICAL	Unsigned applet gains elevated permissions and escapes sandbox; exploited in the wild within 48 hours of disclosure
CVE-2012-1723	2012	CRITICAL	Type-confusion in HotSpot JIT compiler; enables arbitrary code execution outside sandbox — widely used in drive-by attacks
CVE-2012-4681	2012	CRITICAL	Confused Deputy attack enabling full sandbox escape; patch bypass discovered within days
CVE-2013-0422	2013	CRITICAL	Package access check bypass; allows applet to load attacker code with elevated privileges; actively exploited within 24 hours of public disclosure
CVE-2013-2460	2013	CRITICAL	Providers class vulnerability enabling sandbox escape and privilege escalation
CVE-2015-4843	2015	HIGH	Integer overflow enabling memory corruption and sandbox bypass
CVE-2015-2590	2015	CRITICAL	Deserialization vulnerability in Java allowing unauthenticated RCE — class of vulnerabilities that persists in Java server-side to this day

The scale of exploitation was extraordinary: Java Applet-based attacks increased by over 300% in the first half of 2010 alone. By 2013, Java was described by security researchers as "the single biggest source of browser-based malware infections globally." Phrack's definitive paper "Twenty Years of Escaping the Java Sandbox" documents a continuous, unbroken record of sandbox bypass techniques from 1995 to 2015.

4.3 Current Java CVE Position (April 2026)

Oracle continues to release Java CVEs against current supported versions (Java 21, Java 25 LTS). Critically for legacy airport systems, any organisation still running Java 8 JRE to support legacy applet infrastructure is exposed to:

- All CVEs from Java 8 through Java 25 that are unpatched in Java 8 — with no upgrade path if the application requires the applet API (removed in Java 11)
- CVE-2026-22021 — JSSE DoS vulnerability affecting Java SE across multiple versions (April 2026)
- CVE-2026-22003 — HotSpot resource exhaustion affecting sandboxed Java environments
- The complete history of serialization/deserialization vulnerabilities in Java, which remain a persistent attack class

[NOTE] Java 8 extended support ended in March 2022 for commercial users without a paid Oracle support contract. Organisations running Java 8 for legacy applet support are either paying Oracle for extended support (at significant cost under the new per-employee model) or running an unpatched Java 8 JRE — which is equivalent in risk to running IE11 or Chrome 44.

5. Browser Support for Java Applets in 2026

[FACT] As of April 2026, there is no mainstream, supported browser that can run Java Applets. The technology is dead across every current platform.

Browser	Java Applet Support	Notes
Microsoft Edge 147	**NONE**	NPAPI removed; no Java plugin available
Google Chrome 147	**NONE**	NPAPI removed since Chrome 45 (Sep 2015)
Mozilla Firefox 136+	**NONE**	NPAPI removed since Firefox 52 (Mar 2017)

Apple Safari	**NONE**	NPAPI support removed; macOS Java plugin never existed
Internet Explorer 11	ActiveX only — UNSUPPORTED	IE11 with Java 8 ActiveX plugin technically executable but: (a) IE11 unpatched since 2022, (b) Java 8 unpatched without paid Oracle contract, (c) not available on Windows 11
Chrome 44	NPAPI present — CRITICALLY UNSAFE	Last Chrome with NPAPI, but: (a) 103 versions behind, (b) unpatched since 2015, (c) cannot be installed on Windows 11
Pale Moon (fork)	NPAPI supported — UNOFFICIAL	Niche Firefox fork; not a supported enterprise browser; no commercial support; not endorsed by any vendor
Edge IE Mode	ActiveX limited — not full Java	IE Mode does not support full Java ActiveX applet execution; partial compatibility only

The only operational path for running Java Applets in 2026 requires: IE11 (unsupported, unavailable on Windows 11) or Chrome 44 (critically unsafe, cannot be installed on Windows 11). Both of these have been covered in detail in the companion browser comparison report.

[COMPOUNDING RISK] An airline system requiring Chrome 44 + Java Applet via NPAPI represents the worst possible security combination: a browser unpatched for 10 years, with the only remaining NPAPI attack surface, running Java sandbox bypass code with a decade of known exploits. This is not a theoretical risk — it is an active, exploitable attack chain with documented techniques.

6. Oracle Java Licensing — The Commercial Crisis

6.1 The 2023 Licensing Shift

[COMMERCIAL IMPACT] Oracle changed Java SE licensing in January 2023 from per-processor / named-user to per-employee across the entire organisation. Most enterprises saw 3–10x cost increases overnight.

Under the old model, an airline licensing Java SE for 200 DCS terminal workstations at an airport paid for 200 named-user licences. Under the new model, every employee of the airline — including cabin crew, call centre staff, and corporate employees with no interaction with Java whatsoever — counts toward the licence.

Licensing Factor	Old Model (pre-2023)	New Model (Jan 2023+)	Impact
Licence metric	Named User Plus (NUP) or Processor	Per employee — entire organisation	Dramatically larger licence base
Airport terminals (example: 200 workstations)	~200 NUP licences	All airline employees globally	10–30x cost increase typical
Real-world example (12,000 employees)	~$180,000/year	~$2,100,000/year (quoted)	11.7x increase
Part-time / contractor staff	Not counted (typically)	Counted in full	Further inflates costs
Java 8 extended support	Included with SE subscription	Requires separate paid contract	Additional cost layer for legacy Java 8 users
OpenJDK alternative	Free alternative always existed	Free alternative (Temurin, Corretto, etc.)	Migration to OpenJDK eliminates Oracle licensing entirely

6.2 The Licensing Trap for Legacy Aviation Systems

Airlines and airport operators running Java Applet-based DCS or CUSS systems are in a particularly difficult position:

- The legacy application requires Java 8 JRE with browser plugin support — removed in Java 11
- Java 8 extended support requires a paid Oracle Java SE subscription — now priced per-employee
- The application cannot be upgraded to Java 11+ because the applet API was removed

- The only escape is to rebuild the application on modern technology — which is the migration that should have happened in 2015–2018

The commercial options available to these organisations are:

Option	Cost	Security Posture	Recommended?
Continue Oracle Java 8 under new per-employee licensing	Very high — 3–30x previous spend	Partially mitigated (patches available under contract) but applet runtime itself remains vulnerable	NO — unsustainable cost; applet runtime still dangerous
Run unpatched Java 8 without Oracle contract	Zero additional cost	CRITICAL — Java 8 unpatched, applet CVEs unmitigated	NO — indefensible security posture
Migrate to OpenJDK / Eclipse Temurin (Java 8)	Free	Still running Java 8; applet CVEs remain; no Oracle patches	PARTIAL — eliminates licensing cost but does not address security
Rebuild on HTML5 / REST / WebSocket (CUSS 2.0)	One-time development cost	GOOD — eliminates Java Applet attack surface entirely; modern browser compatible	YES — only recommended long-term path
Elevation SDK or managed CUSS 2.0 build	Defined project cost, weeks not months	GOOD — CUSS 2.0 compliant, Edge compatible, fully supported	YES — fastest path to secure, compliant operation

7. Modern Replacements for Java Applet Aviation Applications

Every function previously delivered by Java Applets in DCS, CUTE, CUSS, and ground handling systems can be replicated — and exceeded — using current web standards:

Java Applet Function	Modern Replacement	Benefit
Real-time seat map display	HTML5 Canvas / SVG + WebSocket	Faster rendering, responsive design, no plugin required
DCS check-in form processing	REST API + HTML5 form / React/Vue SPA	Full browser compatibility, TLS 1.3, modern auth (OAuth2)
Boarding card / bag tag printing	WebUSB / Web Serial API + REST	Direct hardware access without Java runtime
Passport / biometric reader integration	WebHID / Web Serial API	Standards-based device access in modern browsers
Barcode / QR scanner input	WebUSB / camera API	Native browser support, no plugin
Real-time flight data feed	WebSocket / Server-Sent Events	Lower latency than Java polling; natively supported in Edge
Weight & balance calculation	JavaScript / WebAssembly	Compiled WASM runs at near-native speed; sandboxed by default
CUTE multi-airline session switching	OAuth2 / OIDC session management	Secure, standards-based identity switching between airline contexts

8. Consolidated Risk Summary

Risk Area	Rating	Key Driver
Security — Java Applet sandbox bypass history	CRITICAL	Decades of documented exploits; sandbox escape reliable and weaponised in exploit kits
Security — Java 8 unpatched (no Oracle contract)	CRITICAL	CVE-2026-22021 and class; no patch without paid per-employee Oracle subscription

Security — NPAPI + Java in Chrome 44	CRITICAL	Compounding risk: 10-year-old browser + Java NPAPI = fully exploitable attack chain
Security — IE11 + Java ActiveX	CRITICAL	Both components unpatched; IE11 unpatched since 2022; not available on Windows 11
Compliance — PCI DSS v4.0	FAIL	Unsupported runtime processing cardholder data; TLS below minimum
Compliance — Cyber Essentials v3.3	FAIL	Unsupported software with known unpatched CVEs cannot meet technical controls
Commercial — Oracle Java SE licensing	HIGH	Per-employee model creates 3–30x cost increase for organisations still on Java SE
Commercial — Cyber insurance	HIGH	Unsupported browser + unpatched Java runtime = grounds for claim denial
Operational — No supported rebuild path	CRITICAL	java.applet package removed from Java 26 (March 2026); cannot compile applet code on current JDK
Operational — No supported browser	CRITICAL	Zero mainstream browsers support NPAPI/Java Applets in 2026
Operational — CUSS 2.0 incompatibility	FAIL	CUSS 1.x discontinued January 2026; Java Applet architecture incompatible with CUSS 2.0 standards

9. Recommendations

9.1 Immediate Actions

- Audit all airline and airport applications to identify any remaining Java Applet dependencies — document browser and JRE version in use
- Assess Oracle Java SE licensing position — determine whether the organisation is inside a paid support contract or running unpatched Java 8
- Isolate Java Applet-dependent systems from internet-facing networks and from systems processing payment or passenger PII data
- Formally register Java Applet applications as end-of-life items in the application portfolio, with an owner and a mandated decommission date
- Review cyber insurance policy wording specifically for language around unsupported software and unpatched runtimes

9.2 Migration Path

- **Target architecture:** Rebuild using HTML5, REST APIs, WebSocket, and OAuth2 — the technology stack required by CUSS 2.0 (effective January 2026). This eliminates Java from the application entirely, removes Oracle licensing exposure, and produces a browser-native application runnable in Edge on Windows 11.
- **Elevation SDK:** The Elevation SDK provides device abstraction, DCS/baggage API integration, and CUSS 2.0 compliant bindings out of the box — reducing the rebuild effort for airline DCS/CUTE/CUSS interfaces from months to weeks.
- **Managed build:** Where internal development capacity is not available, Elevation AI can deliver a complete CUSS 2.0 application — check-in, seat selection, bag tag and boarding card printing, passport scanning — built on Edge-native web standards, deployed and operational in weeks.

- **Ongoing operations:** Pair the new application with the Elevation AI Portal for real-time device monitoring, ops-centre alerting, session analytics, and proactive maintenance scheduling — replacing the reactive break-fix model that characterised the Java Applet era.

Sources: OpenJDK JEP 398 (Deprecate Applet API), JEP 504 (Remove Applet API); Oracle Java SE Support Roadmap; Oracle Java SE Licensing Changes 2023 (SoftwareOne, Redress Compliance); Chromium NPAPI Deprecation Developer Guide; NPAPI Wikipedia; Java Applet Wikipedia; NVD CVE Database (CVE-2012-0507, CVE-2012-1723, CVE-2013-0422, CVE-2015-4843); Phrack — Twenty Years of Escaping the Java Sandbox; Microsoft Security Blog (Java Exploits 2016); IATA CUSS 2.0 Common Use Standards; Oracle Critical Patch Update April 2025/2026.