

Klonbarhetsrevision — sammanfattande rapport

Vad en konkurrent kan läsa ut ur er app

Mattermost Desktop 6.3.1 — statisk analys av klienten

Objekt	Mattermost Desktop 6.3.1 (Electron, Windows x64)
Beställd av	Internt övningsuppdrag (ingen extern beställare)
Utförd av	Axel Sjöberg, Kodro
Datum	16 oktober 2026
Klassificering	Konfidentiellt

Övningsrapport, byggd på en öppen app (Mattermost Desktop, Apache-2.0). Ingen kund har beställt analysen. En skarp kundrapport levereras krypterat och endast till beställaren.

Om Kodro

Kodro utför klonbarhetsrevisioner: en oberoende granskning av vad som i praktiken går att läsa ut ur en levererad applikation med dagens AI-stödda analysverktyg. Granskningen görs statiskt på exakt den fil som kunden distribuerar, med kundens skriftliga tillstånd, och mynnar ut i en konkret, åtgärdbar rapport.

Syftet är defensivt: att visa en tillverkare vad en konkurrent eller en motiverad tredjepart kan utvinna ur produkten — källkod, affärslogik, oannonserade funktioner och eventuella inbäddade hemligheter — innan någon annan gör det. Kodro angriper aldrig produktionssystem och använder aldrig funna hemligheter.

Tjänst	Klonbarhetsrevision (statisk analys av levererad klient)
Utförs av	Axel Sjöberg, grundare
Kontakt	kontakt@kodro.se · kodro.se

Meddelanden och förbehåll

Upphovsrätt och spridning

© 2026 Kodro. Rapporten är konfidentiell och får inte reproduceras eller spridas utan skriftligt medgivande. Detta exemplar är ett internt övningsuppdrag på en öppen app (Mattermost Desktop-klienten, Apache-2.0) och har inte beställts av någon kund. Mattermosts server har blandad licens (AGPL för gratisdelen, kommersiell för Enterprise); endast desktop-klienten analyserades, och serverns Enterprise-delar behandlas inte som öppen källkod.

Täckningsförbehåll

Granskningen är tidsbegränsad och bygger enbart på statisk analys av den levererade filen vid en viss tidpunkt. Den är inte en uttömmande förteckning över allt som kan utläsas ur produkten, och frånvaron av ett fynd är inte en garanti för att motsvarande problem inte finns. Resultatet gäller den analyserade versionen; en ny release kan införa nya läckor och bör granskas på nytt.

Oberoende

Kodro har inget ägande i eller annan koppling till det granskade projektet. Analysen utfördes på en publikt tillgänglig, verifierad officiell release. Skulle en verklig, utnyttjbar sårbarhet hittas rapporteras den först ansvarsfullt till Mattermost, och inget publiceras förrän den är åtgärdad.

Innehåll

Om Kodro	2
Meddelanden och förbehåll	3
Uppdragssammanfattning	5
Analysobjekt	6
1 Sammanfattning för ledningen	7
2 Översikt över fynd	8
3 Detaljerade fynd	9
4 Åtgärdsplan	12
5 Rekommenderad uppföljning	13
Bilaga A Skalar och kategorier	14
Bilaga B Metod i detalj	15

Uppdragssammanfattning

Kontaktuppgifter

Granskare	Axel Sjöberg — grundare, Kodro · kontakt@kdro.se
Rapportansvarig	Axel Sjöberg

Tidslinje

16 okt 2026	Officiell release hämtad från github.com/mattermost/desktop och hash beräknad
16 okt 2026	Statisk analys genomförd; klontest ~13 min
16 okt 2026	Rapport sammanställd och levererad

I ett skarpt uppdrag löper tidslinjen normalt över flera dagar, med en avstämning efter leverans och en möjlig omgranskning efter åtgärd.

Analysobjekt

Granskningen omfattar en (1) analysenhet: den officiella skrivbordsklienten för Windows.

Objekt	Mattermost Desktop
Version	6.3.1
Fil	mattermost-desktop-6.3.1-win-x64.zip (151 414 592 byte)
Verifiering	SHA-256 6fff38bd...57294 (egen beräkning; releasen publicerar ingen Windows-checksumma)
Typ	Electron-applikation (app.asar)
Plattform	Windows x64
Källa	github.com/mattermost/desktop (officiell release)
Licens	Apache-2.0 (desktop-klienten)

Mattermosts servrar, API och konton samt serverns Enterprise-delar ligger utanför omfattningen och behandlas inte som öppen källkod.

1 Sammanfattning för ledningen

Klonbarhet A	Exponering Låg	Klontest ~13 min
------------------------	--------------------------	----------------------------

Engagemangsöversikt

Kodro granskade den levererade Mattermost-klienten statistiskt för att fastställa hur mycket av produkten som kan förstås och klonas enbart ur desktop-appen: hur den kopplar upp mot servern, autentisering och tokenhantering, samt eventuell lokal logik utöver skalet.

Observationer och påverkan

Mattermost Desktop går inte att kлона som produkt utifrån klienten (Klonbarhet A). Appen är ett Electron-skal vars enda uppgift är att ladda serverns webbapp i ett fönster och koppla in operativsystemet (notiser, skärmdelning, nedladdningar, säker tokenlagring). All affärslogik — chatt, kanaler, behörigheter, licensiering, prissättning — ligger på servern och levereras i runtime.

En konkurrent som bara har desktop-appen får veta hur skalet är byggt (vilket ändå är publik open source) men ingenting om själva produkten. Det allvarligaste fyndet är en Sentry-DSN i klartext (F01), vars påverkan är försumbar: den är write-only och ger ingen data- eller kontoåtkomst. Auth-token ligger inte i paketet utan hämtas via operativsystemets säkra lagring.

Detta är avsiktligt ett A-exempel på en skyddande arkitektur: hög klonbarhet på skalet spelar ingen roll när affärsvärdet ligger server-side. Att klienten dessutom är open source gör klonbarheten irrelevant som risk.

Kritisk	0
Hög	0
Medel	0
Låg	1
Info	2

Rekommendationer

1. Ingen brådskanie åtgärd. Roter eller filtrera Sentry-DSN:en om krasch-spoofing oroar (F01).
2. Behåll den nuvarande arkitekturen — server-side-logik plus tunt skal — vid framtida funktioner.
3. Bibehåll säkerhetshygienen i skalet: kontextisolering, contextBridge och ingen nodeIntegration.

2 Översikt över fynd

<i>ID</i>	<i>Titel</i>	<i>Typ</i>	<i>Allvarlighet</i>
F01	Sentry-DSN inkompilerad i klienten	Hemlighet	Låg
F02	Hela desktop-/IPC-API-ytan kan kartläggas ur klienten	Interna API:er	Info
F03	Inga dolda funktioner; developer mode synligt, feature flags serverstyrda	Dold funktion	Info

3 Detaljerade fynd

F01 Sentry-DSN inkompilerad i klienten

Allvarlighet	● Låg	Typ	Hemlighet
Åtgärdsinsats	Liten	Fynd-ID	F01
Plats	resources/app.asar → index.js (huvudprocess, Sentry-init)		

Beskrivning

Huvudprocessen initierar Sentry med en DSN i klartext i den buntade koden. En Sentry-DSN är konstruerad för att följa med klienter och ger bara rätt att skicka händelser (write-only), inte läsa data. Den är alltså inte en känslig produktionsnyckel, men den identifierar Mattermosts interna telemetri-projekt och går att återanvända. `sendDefaultPii` är satt till `false`.

Belägg (*maskerat*)

```
Qc({
  dsn: "https://cdba...4ec7@o94110.ingest.us.sentry.io/4510...8688",
  sendDefaultPii: !1, // false
  environment: e ? "prerelease" : "stable",
  attachStacktrace: !0
})
```

Påverkan för verksamheten

Begränsad. En konkurrent eller attackerare som extraherar DSN:en kan skicka påhittade krasch-/felhändelser till Mattermosts Sentry-projekt och därmed skapa brus eller förbruka kvot. Ingen data kan läsas ut och inga konton påverkas. Läckaget avslöjar bara att produkten använder Sentry för felrapportering — ingen affärslogik.

Rekommendation

1. Ingen rotation krävs av säkerhetsskäl (write-only). Om brusrisken ska stängas: aktivera inbound-filter/rate limiting i Sentry och överväg roterad DSN.
2. Om telemetri-endpointen inte ska vara publik, proxa Sentry-events via egen backend i stället för att bädda in DSN i klienten.

F02 Hela desktop-/IPC-API-ytan kan kartläggas ur klienten

Allvarlighet ● **Info**

Typ Interna API:er

Åtgärdsinsats Liten

Fynd-ID F02

Plats `resources/app.asar → externalAPI.js, internalAPI.js (preload), index.js`

Beskrivning

Klientens kompletta kommunikationsyta går att läsa direkt ur paketet: två contextBridge-bryggor (desktopAPI, desktop) och ~143 namngivna IPC-kanaler. Allt innehåll är desktop-plumbing — fönster-/flik-/serverhantering, notiser, calls/skärmledning, nedladdningar, tema-synk, uppdateringskontroll och säker tokenlagring. Ingen prissättning, licenskontroll, validering eller proprietär algoritm finns i klienten. Auth-token hämtas via secure-storage-get (OS:ets säkra lagring) och ligger inte i paketet.

Belägg (maskerat)

```
e.contextBridge.exposeInMainWorld("desktop", {
  getSecret: (n, r) => e.ipcRenderer.invoke("secure-storage-get", n, r),
  isDeveloperModeEnabled: () => e.ipcRenderer.invoke("is-developer-mode-enabled"),
  addServer: n => e.ipcRenderer.send("add-server", n),
  validateServerURL: (n, r) => e.ipcRenderer.invoke("validate-server-url", n, r),
  getConfiguration: () => e.ipcRenderer.invoke("get-configuration"),
  // ... server-/flik-/nedladdnings-/tema-kanaler, inga affärshemligheter
});
```

Påverkan för verksamheten

Ingen direkt risk. En konkurrent lär sig exakt hur desktop-skalet är byggt — men det är redan publik open source (Apache-2.0), så inget hemligt exponeras. Att hela IPC-ytan är läsbar underlättar inte kloning av själva produkten, eftersom chatt- och samarbetslogiken ligger på servern.

Rekommendation

1. Ingen åtgärd krävs — detta är förväntat för en open source-klient.
2. Behåll contextBridge och kontextisolering och fortsätt validera senderFrame/ursprung för känsliga IPC-anrop.

F03 Inga dolda funktioner; developer mode synligt, feature flags serverstyrda

Allvarlighet ● **Info**

Typ Dold funktion

Åtgärdsinsats Liten

Fynd-ID F03

Plats `resources/app.asar → index.js, common/config/buildConfig.js`

Beskrivning

Sökning efter dolda vyer, admin-vägar, odokumenterade feature flags och developer-mode-bakdörrar gav inget. Developer mode existerar men är användarstyrkt: valen ligger öppet i Visa-menyn, sparas i `developerMode.json`, och en synlig skärmindikator visas när läget är på. Produktens feature flags läses från serverns konfiguration i runtime, inte från gömd kod i klienten. Byggkonfigurationen är dokumenterad och utan hemligheter.

Belägg (maskerat)

```
var buildConfig = {
  defaultServers: [ /* (tomt) */ ],
  enableServerManagement: true,
  updateNotificationURL: 'https://releases.mattermost.com/desktop',
  allowedProtocols: ['mattermost', 'ftp', 'mailto', 'tel'],
};
// i18n: "Developer mode is enabled. You should only have this enabled if a
Mattermost[...]"
```

Påverkan för verksamheten

Ingen. Avsaknaden av dolda funktioner är ett positivt resultat: det finns inget odokumenterat i klienten som läcker roadmap eller intern funktionalitet.

Rekommendation

1. Ingen åtgärd krävs.

4 Åtgärdsplan

Omedelbart

- Inget kritiskt. Beslut: ska Sentry-DSN:en (F01) filtreras/roteras? Standardrekommendation: lågprioriterat.

Kort sikt

- Om telemetry-brus är en reell oro: aktivera inbound-filter/rate limiting i Sentry, eller proxa events via egen backend (F01).

Strukturellt

- Behåll arkitekturen tunt skal plus server-side-logik för nya funktioner — det är den som ger Klonbarhet A.
- Behåll säkerhetshygienen i Electron-skalet (kontextisolering, contextBridge, ingen nodeIntegration, användarstyrd cert-trust).

5 Rekommenderad uppföljning

Kör en ny analys efter nästa större desktop-release, eller kvartalsvis, för att fånga eventuell drift — t.ex. om ny affärslogik av misstag flyttas in i klienten eller om en känslig nyckel bäddas in. Så länge logiken ligger server-side förblir klonbarheten A oavsett hur mycket skalet analyseras.

Bilaga A — Skalar och kategorier

Klonbarhet och exponering mäter olika saker och vägs var för sig. Klonbarheten anger hur snabbt kärnfunktionen kan återskapas; exponeringen anger det allvarligaste enskilda fyndet.

Klonbarhetsskala

Nivå	Klontest	Innebörd
A	Går inte	Kärnlogiken ligger på servern och kan inte rekonstrueras ur klienten.
B	Mer än en arbetsvecka	Kräver betydande dekompilering och egen återuppbyggnad.
C	En till fem arbetsdagar	Logiken är läsbar men utspridd; kräver sammanställning.
D	En till åtta timmar	Stora delar kan läsas direkt; begränsat återställningsarbete.
E	Under en timme	Koden kan läsas nästan ordagrant; specifikationen faller nästan ut direkt.

Exponeringsnivåer

Nivå	Innebörd
 Kritisk	Inbäddad produktionshemlighet med omedelbar ekonomisk eller säkerhetsmässig konsekvens (giltig nyckel, token, lösenord).
 Hög	Känslig affärslogik eller hemlighet som ger en konkurrent betydande och direkt fördel.
 Medel	Exponering som bör åtgärdas men utan omedelbar hemlighetsförlust (t.ex. medskickad källkod eller source maps).
 Låg	Avläsbar information av begränsat värde, ofta avsiktligt öppen eller svår att utnyttja.
 Info	Observation utan direkt risk; noteras för fullständighet eller som positivt fynd.

Fyndkategorier

Kategori	Innebörd
Källkodsexponering	Original- eller nära-originalkällkod är läsbar ur den levererade filen.
Roadmap-läckage	Oannonserade funktioner, flaggor eller planer syns i bygget.
Affärslogik	Intern logik, protokoll eller scheman kan kartläggas ur klienten.
Dold funktion	Funktionalitet som finns i koden men inte är exponerad för användaren.
Build-hygien	Byggartefakter (t.ex. source maps) följer oavsiktligt med i releasen.
Interna API:er	Klientens interna gränssnitt (IPC, bryggor) kan kartläggas ur paketet.
Hemlighet	Nycklar, tokens eller motsvarande — eller bekräftad frånvaro av sådana (positivt fynd).

Bilaga B — Metod i detalj

Analysen utfördes lokalt och defensivt. Endast statistisk analys av den levererade filen. Ingen nätverkskontakt mot Mattermosts servrar, API eller konton, och inga funna värden verifierades mot någon tjänst.

1. **Anskaffning och verifiering.** Målfilen hämtades som officiell open source-release från github.com/mattermost/desktop och verifierades med egen beräknad SHA-256 (releasen publicerar ingen Windows-checksumma).
2. **Uppackning.** Electron-paketet packades upp med asar och app.asar extraherades lokalt.
3. **Statisk applikationsanalys.** Genomgång av huvudprocess, preload-bryggor och IPC-kanaler samt byggkonfiguration.
4. **Söksvep.** Riktade sökningar efter hemligheter, endpoints och feature flags i den buntade koden.
5. **Arkitekturbedömning.** Fastställande av att kärnlogiken laddas från servern i runtime och inte finns i klienten.

Avgränsningar

Analysen är statisk — ingen kod kördes och ingen trafik observerades. Native-tillägget koffi.node (och fem andra .node-binärer) hade strippad/omsignerad integritet efter paketering och behandlades som otillförlitliga (godartat, vanligt för Electron-bygget). Den semantiska grafen är partiell (resurstak på de största rendererbundlarna); statistiska fakta och modulrelationer är fullständiga.