**Co osiągniemy**

* **Minimal-PII telemetry z frontu**: brak cookies/storage ( sendCredentials:false, centralConfig:false ), obcięte ?query/#hash w URL-ach, bez nagłówków w payloadzie, whitelist śledzenia rozproszonego tylko do własnych domen.
* **Brak IP/UA w indeksach**: final ingest-pipeline w Kibanie usuwa client.ip, source.ip, user\_agent i nagłówki — **PII nie trafia do Elasticsearch** (ani do backupów od momentu wdrożenia).
* **Mniejszy wolumen i „widoczność” danych**: sampling (transactionSampleRate) i/lub rate-limit → mniej rekordów o użytkownikach.
* **Lepsza zgodność z zasadami RODO**: szczególnie **minimalizacja** i **privacy by default**; łatwiejsza argumentacja dla **uzasadnionego interesu** (brak trwałych identyfikatorów, tylko agregaty wydajności).
* **Mniej pracy przy żądaniach dostępowych/erasure**: skoro nie przechowujemy IP/UA i treści nagłówków, zakres przeszukiwania jest dużo mniejszy.
* **Mniejsze ryzyko wycieku PII przez URL**: referrery i adresy „oczyszczone” przed wysyłką.

**Jakie ryzyka pozostaną (i na co uważać)**

* **Warstwa sieciowa**: APM endpoint i po drodze (proxy/WAF) *chwilowo* zobaczą IP i Referer. Jeśli to problem, dodaj dla ścieżki intake nagłówek **Referrer-Policy: no-referrer** w serwerze/proxy i nie loguj IP/UA na poziomie reverse proxy.
* **PII w custom danych**: własne etykiety (labels) lub komunikaty błędów mogą przypadkiem zawierać e-mail/ID. Trzeba trzymać dyscyplinę w kodzie i zostawić filtr „ostatniej szansy” (apm.addFilter).
* **Konfiguracja śladu**: błędna lista distributedTracingOrigins może roznieść nagłówki trace do domen trzecich.
* **Regulacyjne „szare strefy”**: nawet bez cookies, niektórzy regulatorzy mogą kwestionować podstawę **uzasadnionego interesu** przy szerokim, stałym monitoringu. Warto mieć LIA/DPIA, informację w polityce prywatności i rozsądne retencje.
* **Dziedziczone dane**: stare indeksy/backupy sprzed wdrożenia mogą już zawierać IP/UA — pipeline nie działa wstecz.
* **Zmiany wersji/konfiguracji**: aktualizacje integracji APM lub nowe data-streams mogą ominąć pipeline, jeśli nie zrobisz go „final pipeline” dla wszystkich odpowiednich strumieni.
* **Fingerprinting pośredni**: nawet bez cookies, kombinacja UA/timing może teoretycznie umożliwiać pseudonimizację; minimalizacja zakresu i sampling ograniczają ryzyko, ale go nie zerują.
* **Dostępy operacyjne**: nadal obowiązuje kontrola dostępu do Kibany/ES, szyfrowanie w spoczynku, retencje i logowanie zmian — to poza RUM, ale krytyczne dla zgodności.

**Wniosek**

Po wdrożeniu tych zmian będziesz zbierać **użyteczne metryki wydajności** przy **maksymalnej minimalizacji danych osobowych**: nic trwałego w przeglądarce, brak IP/UA w ES, oczyszczone URL-e i ograniczona propagacja śladu. Pozostają ryzyka tranzytowe (sieć/proxy), organizacyjne (dostępy, retencja) i prawne (LIA/DPIA), ale są one **zarządzalne** dobrymi praktykami operacyjnymi i polityką prywatności.

Da się sporo zrobić „od ręki” z poziomu **front-endu (Angular/RUM)** i **Kibany**, nawet bez dostępu do APM Servera. Poniżej konkretne kroki i gotowce.

**1) Z frontu (Angular + RUM JS) — wysyłaj mniej i bez PII**

Ustawienia startowe (bez cookies/storage, bez propagacji do obcych domen, sampling) + filtr wycinający PII z URL-i/spans:

// apm.ts

import { init as initApm } from '@elastic/apm-rum';

export function bootApm() {

// Szanuj DNT i włącz sampling tylko dla części ruchu

const dnt = (navigator.doNotTrack === '1' || (window as any).doNotTrack === '1');

if (dnt) return null; // nie ładuj APM wcale

const apm = initApm({

serviceName: 'frontend-angular',

serverUrl: 'https://<twój-endpoint-apm-or-proxy>', // może prowadzić przez reverse proxy

environment: 'production',

// prywatność

centralConfig: false, // nie używaj sessionStorage do central config

sendCredentials: false, // nie wysyłaj cookies do APM

distributedTracing: true,

distributedTracingOrigins: ['https://api.twojadomena.pl'], // tylko własne API

propagateTracestate: false, // nie propaguj dodatkowych nagłówków do innych originów

// kontrola wolumenu

transactionSampleRate: 0.2, // 20% transakcji (dobierz do ruchu)

// jeśli jakieś auto-instrumentacje są potencjalnie „gadatliwe”, można je wyłączyć:

// disableInstrumentations: ['click'] // przykładowo

});

// Redakcja PII z payloadu zanim opuści przeglądarkę

apm.addFilter((payload: any) => {

// transakcje: oczyść page.url i referer

payload.transactions?.forEach((t: any) => {

const page = t.context?.page;

if (page?.url) {

try {

const u = new URL(page.url);

page.url = u.origin + u.pathname; // bez ?query i #hash

} catch { delete page.url; }

}

if (page?.referer) page.referer = '';

// spans: obetnij PII z URL-i http

t.spans?.forEach((s: any) => {

const h = s.context?.http;

if (h?.url) {

try {

const u = new URL(h.url);

u.search = ''; u.hash = '';

h.url = u.origin + u.pathname;

} catch { delete h.url; }

}

// wytnij nagłówki, jeśli by się pojawiły

if (h?.response?.headers) delete h.response.headers;

if (h?.request?.headers) delete h.request.headers;

});

});

// możesz całkowicie porzucić wydarzenie, jeśli wykryjesz PII:

// if (foundPII(payload)) return false;

return payload;

});

return apm;

}

Kluczowe opcje i ich **bezpieczne domyślne** wg dokumentacji RUM: sendCredentials = false, centralConfig = false, a apm.addFilter pozwala redagować/usuwać pola przed wysyłką. Ograniczaj propagację śladu do **konkretnych** własnych domen przez distributedTracingOrigins. Użyj transactionSampleRate, by zmniejszyć wolumen danych. ([Elastic](https://www.elastic.co/docs/reference/apm/agents/rum-js/configuration" \o "Configuration | APM RUM JavaScript agent))

Uwaga: sama **prośba HTTP do APM** nadal technicznie zawiera IP/UA (warstwa sieci). Poniżej pokazuję, jak sprawić, by **nie trafiły do indeksu** w Elasticsearch. To różnica między „co serwer widzi chwilowo” a „co jest trwale przechowywane”. RODO dotyczy m.in. IP jako danych osobowych (Motyw 30). ([GDPR](https://gdpr-info.eu/recitals/no-30/?utm_source=chatgpt.com))

**2) Z Kibany — zredaguj/usuń IP, UA i inne pola na ingest pipeline**

Skoro nie możesz zmienić APM Servera, **utwórz własny pipeline ingest** i podłącz go jako „final pipeline” do strumieni danych APM (transakcje/spany/błędy). Użyj procesorów **remove** oraz (opcjonalnie) **redact**:

PUT \_ingest/pipeline/apm\_scrub\_pii

{

"description": "Usuwa PII z dokumentów APM",

"processors": [

{ "remove": { "field": ["client.ip","source.ip","user\_agent","user\_agent.\*"], "ignore\_missing": true }},

{ "remove": { "field": ["http.request.headers","http.response.headers","url.query"], "ignore\_missing": true }},

{ "remove": { "field": ["context.request.headers","context.response.headers"], "ignore\_missing": true }},

{ "redact": { "field": "message", "patterns": ["%{EMAILADDRESS}","%{IP}"], "replacement": "<REDACTED>", "ignore\_missing": true } }

]

}

Następnie **podłącz pipeline** do strumieni APM w Kibanie:

* jeśli korzystacie z **APM Integration przez Fleet**: w Kibanie → *Integrations* → *Elastic APM* → *Edit integration* → sekcja **Processors**/**Ingest pipelines** (lub włącz „Capture personal data: false”, jeśli dostępny toggl w waszej wersji). ([Elastic](https://www.elastic.co/guide/en/apm/server/7.15/input-apm.html?utm_source=chatgpt.com))
* alternatywnie: *Stack Management* → *Data Streams* → wybierz strumienie APM i ustaw stworzony pipeline jako **final pipeline** (działa dla data streams APM). ([Elastic](https://www.elastic.co/docs/solutions/observability/apm/data-streams?utm_source=chatgpt.com))

Dokumentacja procesorów: **remove** (usuwa wskazane pola) i **redact** (maskuje wzorce, np. e-mail/IP). ([Elastic](https://www.elastic.co/docs/reference/enrich-processor/remove-processor?utm_source=chatgpt.com))

**3) (Opcjonalnie) Reverse proxy dla pełnej kontroli**

Jeśli masz jakikolwiek serwer www pod swoją kontrolą, ustaw serverUrl na **własny** endpoint (np. /rum/intake) i przekaż dalej do APM Servera. Tam możesz zdjąć/zmienić nagłówki User-Agent/X-Forwarded-For, a także *przemapować* ścieżkę (ułatwia też obejście adblocków blokujących /rum/events). Proxy musi przekierować dalej na właściwe /intake/v{apiVersion}/rum/events. ([Elastic](https://www.elastic.co/docs/reference/apm/agents/rum-js/configuration), [Stack Overflow](https://stackoverflow.com/questions/68723140/post-https-apm-acme-com-intake-v2-rum-events-neterr-blocked-by-client?utm_source=chatgpt.com))

**4) Szybka „lista kontroli”**

* W kodzie: centralConfig:false, sendCredentials:false, dopracowane distributedTracingOrigins, transactionSampleRate ≤ 0.2, apm.addFilter tnący query/hash i nagłówki. ([Elastic](https://www.elastic.co/docs/reference/apm/agents/rum-js/configuration))
* W Kibanie: final pipeline apm\_scrub\_pii podpięty do strumieni APM lub toggle **Capture personal data = false** (jeśli dostępny). ([Elastic](https://www.elastic.co/docs/solutions/observability/apm/built-in-data-filters?utm_source=chatgpt.com))
* W Discover/KQL sprawdź, że not exists: client.ip and not exists: user\_agent dla nowych dokumentów.
* Polityka prywatności z informacją o RUM (podstawa prawna, zakres, retention).

Jeśli chcesz, przygotuję dla Ciebie:

* minimalny moduł Angular (provider), który ładuje RUM **warunkowo** (np. z DNT / consentem) i inicjuje apm jak wyżej,
* „klikany” JSON do Dev Tools w Kibanie, który podłączy pipeline jako „final pipeline” do właściwych data streams.