COMPLIANCE · TEMPLATE

Securing PHI in *Observability* Data

KAANSYSTEMS.COM/LIBRARY/SECURING-PHI-OBSERVABILITY · MAY 30, 2026

ABOUT THIS TEMPLATE

Logs, traces, and metrics frequently capture protected health information by accident. How to keep your monitoring stack out of HIPAA breach territory.

THE TEMPLATE

A scrubber config with regex patterns for the most common PHI types. Apply at the observability-agent layer (format shown for Datadog's Sensitive Data Scanner; patterns are vendor-independent).

FIELD	PATTERN	REPLACEMENT
SSN	`\b\d{3}-\d{2}-\d{4}\b`	`[REDACTED-SSN]`
SSN (no dashes)	`\b\d{9}\b` (context-dependent)	`[REDACTED-SSN]`
MRN (common formats)	`\b(MRN[-:\s]*\d{6,10}	\d{8})\b`
DOB	`\b(0[1-9]	1[012])[-/](0[1-9]
Email	`\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b`	`[REDACTED-EMAIL]`
Phone	`\b(\+\d{1,2}\s?)?\(?\d{3}\)?[\s.-]?\d{3}[\s.-]?\d{4}\b`	`[REDACTED-PHONE]`
Credit card	`\b(?:\d{4}[-\s]?){3}\d{4}\b`	`[REDACTED-CC]`
Auth bearer token	`Bearer\s+[A-Za-z0-9._~+/=-]+`	`Bearer [REDACTED]`
API key (generic)	`(sk	api)[-_][A-Za-z0-9]{20,}`

For Datadog specifically, this lives at `Logs → Configuration → Pipelines → Sensitive Data Scanner`. The same patterns apply (sometimes with vendor-specific escape rules) to:

- **Elastic ingest pipelines** (`gsub` processors)
- **New Relic log obfuscation rules**
- **Splunk SEDCMD transforms**
- **Cloudflare Logpush filters**

Test the patterns against your actual logs before turning them on in production. Start in "tag-only" mode where matches are flagged but not modified, then flip to "replace" once you've verified no false-positive damage to debugging.

HOW TO USE IT

Add an audit step to your engineering checklist for new PHI-touching code: before merge, search the diff for likely log-leak candidates. A regex for `logger.info` or `console.log` followed by a variable reference catches most of them.

You will have leaks. Recovery is to immediately rotate the scrubber pattern to cover the new shape, then run the storage-layer access audit for the period the leak was live. If no one accessed the affected logs, you have a finding to document. If someone did, you have a more serious finding to escalate.

The hardest leak to catch is the structured one: a developer adds an `extra={'patient_id': ...}` field to a log call intending it to be opaque, but the field naming convention propagates patient identifiers into the index. Pattern-based scrubbing won't catch that. Only code review will.