Read History: Monitoring document access for GDPR compliance

Jochen Kressin · 2018-05-30

Under GDPR, providing access to *"personally identifiable information" (PII)* has become a sensitive topic. PII data is all data that can be associated directly with a person, like name, address, email or under some circumstances even IP addresses. GDPR aims to give the control over what PII data can be stored and for what it can be used back to the owner of the data. This means you cannot simply give anyone access to this kind of data anymore, but need to implement tight security and audit measures. The Search Guard Read History helps you to monitor access to PII data and stay compliant with GDPR.

Giving back control over personal data

GDPR is a heavy burden on anyone who stores personal information of customers and users. In particular, it mandates that the owner of the data shall have full control over what happens with his or her data. Data must only be processed for "intended purposes", and the data owner can demand changes to these purposes at any time. In addition, the data owner can demand information about what persons had or have access to the data, and for what purpose(s) they accessed it. Failing to provide this data to the customer can lead to high fines. So what is the solution?

Tracking data access in Elasticsearch - down to the field level

Data access in Elasticsearch cannot really be tracked, neither with Elasticsearch out-of-the-box, nor with any additional plugins or features at the time of writing. With audit logging you are able to monitor which queries have been executed against a particular index, but you do not get any information about what documents and fields were included in the result set.

Here's where the Search Guard Read History comes to the rescue. It makes it possible to track exactly which documents have been accessed by what user, and which fields were included in the result set.

Why field level is relevant

Why is it important to track access al the way down to the field level. Isn't it enough to just track access to the documents? The answer is quite simple. Not all fields are PII relevant, and fields can be filtered or anonymized:

- Fields can be removed from documents by adding a source filter to the query
- Fields can be removed by using Search Guard Field Level security
- Field can be anonymized by using the Search Guard field anonymization feature, rendering access irrelevant for GDPR

So even if an employee queries PII relevant documents, the result does not necessarily contain PII relevant fields. The Search Guard Read History analyzes the result and only generates an compliance event if the result set actually contains PII fields. This means bullet-proof audit events for GDPR.

Beyond GDPR

Being able to track field level access is not only useful for GDPR compliance. In fact there are a plethora of use cases where this feature is extremely helpful. A company I once worked for faced a data breach where email addresses of customers had been illegally accessed and then sold to some shady marketing companies. As you can guess this caused major problems and a loss in customer trust. After first investigation it became clear that this was an attack from the inside, means by an employee or contractor. However, it was not possible to clearly relate the breach to one or more individuals. By tracking and recording read access to the email field this would have been a no brainer.

PII data in Elasticsearch

Let's have a look at some hypothetical PII relevant data in an Elasticsearch index called *customers*:

```
{
 "FirstName": "PETER",
 "LastName": "MILLER",
 "Email": "petermiller@gmail.com",
 "Address": "34 River St.Chapel Hill, NC 27516"
}
```

The first step in using the Search Guard Read History is to specify

- what indices should be tracked for read access
- what fields in the configured indices should be tracked for read access

In our example, the index is called *customers* and the we want to track *FirstName*, *LastName*, *Email* and *Address*. The corresponding configuration entry in elasticsearch.yml thus looks like:

```
searchguard.compliance.history.read.watched_fields:
  - customers,FirstName,LastName,Email,Address
```

We can also use wildcards in both the index and the field definitions, so this configuration would be equivalent with the configuration above:

```
searchguard.compliance.history.read.watched_fields:
 - customers,*Name,Email,Address
```

Tip: If not really necessary try to void wildcards. Listing all fields individually will give you a slight performance benefit over using wildcards.

Accessing PII data

So let's try to do execute a simple search on the said index, and see what read events are recorded. In our case, we query for a user's record by email with a Search Guard user *hr*employee_:

```
curl -Ss -u hr_employee:hr_employee -H 'Content-Type: application/json' -XPOST
"https://example.com:9200/customers/_doc/_search?pretty" -d \
'{
 "query" : {"term" : {"Email":"petermiller@gmail.com"}}
}'
```

Which will, unsurprisingly, return the users record:

```
{
 "hits": [{
 "_index": "customers",
 "_type": "_doc",
 "_id": "1",
 "_score": 0.2876821,
 "_source": {
 "FirstName": "PETER",
 "LastName": "MILLER",
 "Email": "petermiller@gmail.com",
 "Address": "34 River St.Chapel Hill, NC 27516"
 }
 }]
}
```

Since we did not apply any filtering, all fields in this document have been returned in the result set, so the corresponding read event should list all of them. A simple query on the audit log index returns the following read event:

```
curl -Ss -u admin:admin -H 'Content-Type: application/json' -XGET
https://example.com:9200/auditlog-docread/auditlog/_search?pretty -d \
'{
 "query": {
   "match": {
     "audit_category": {
        "query": "COMPLIANCE_DOC_READ"
     }
   }
 },
 "sort": [{
    "audit_utc_timestamp": {
    "order": "desc"
    }
 }]
}'
```

```
 {
 "_index": "auditlog-docread",
 "_type": "auditlog",
 "_source": {
    ...
    "audit_category": "COMPLIANCE_DOC_READ",
    "audit_request_body": "
{\"Email\":\"petermiller@gmail.com\",\"LastName\":\"MILLER\",\"Address\":\"34 River St.Chapel
Hill, NC 27516\",\"FirstName\":\"PETER\"}",
    "audit_utc_timestamp": "2018-05-15T12:37:21.350+00:00",
    "audit_request_remote_address": "172.16.0.254",
    "audit_trace_doc_id": "1",
    "audit_node_host_address": "172.16.0.2",
    "audit_request_effective_user": "hr_employee",
    "audit_trace_resolved_indices": [
      "customers"
    ],
    ...
 }
}
```

The *auditrequestbody* lists all PII fields that the user *hr*employee_ has accessed. Let's now try to filter some of the fields, and see how this reflects in the READ event.

```
curl -Ss -u hr_employee:hr_employee --insecure -H 'Content-Type: application/json' -XPOST
"https://sgssl-0.example.com:9200/customers/_doc/_search?pretty" -d \
'{
   "_source" : ["Email", "LastName"],
   "query" : {"term" : {"Email":"petermiller@gmail.com"}}
}'
```

We only include the fields *Email* and *LastName* in the result, all other fields are filtered. Consequently, the READ event will list only the *Email* and *LastName* fields:

```
 {
 "_index": "auditlog-docread",
 "_type": "auditlog",
 "_source": {
    ...
    "audit_category": "COMPLIANCE_DOC_READ",
    "audit_request_body" : "{"Email":"petermiller@gmail.com","LastName":"MILLER"}",
    "audit_utc_timestamp": "2018-05-15T12:37:21.350+00:00",
    "audit_request_remote_address": "172.16.0.254",
    "audit_trace_doc_id": "1",
    "audit_node_host_address": "172.16.0.2",
    "audit_request_effective_user": "hr_employee",
    "audit_trace_resolved_indices": [
      "customers"
    ],
    ...
 }
}
```


Summary

In this article we demonstrated how you can use the Search Guard Read History feature to track access to documents in fields in your Elasticsearch cluster. We generated events that includs:

- What document has been accessed
- When the document has been accessed
- What user has accessed the document
- What PII relevant fields were included in the result

This helps to stay compliant especially with the informational rights of customers under GDPR.

Where to go next

- Download the Search Guard Compliance Edition and try the READ history for yourself
- Learn about all Search Guard Compliance features in the official documentation
- Combine the Write History with the Immutable Indices feature to ensure data integrity

Image: shutterstock / stickerama

This article was published on the Search Guard blog: https://search-guard.com/blog/read-history-gdpr/
Search Guard is Security and Alerting for Elasticsearch and Kibana — search-guard.com · Start a free trial