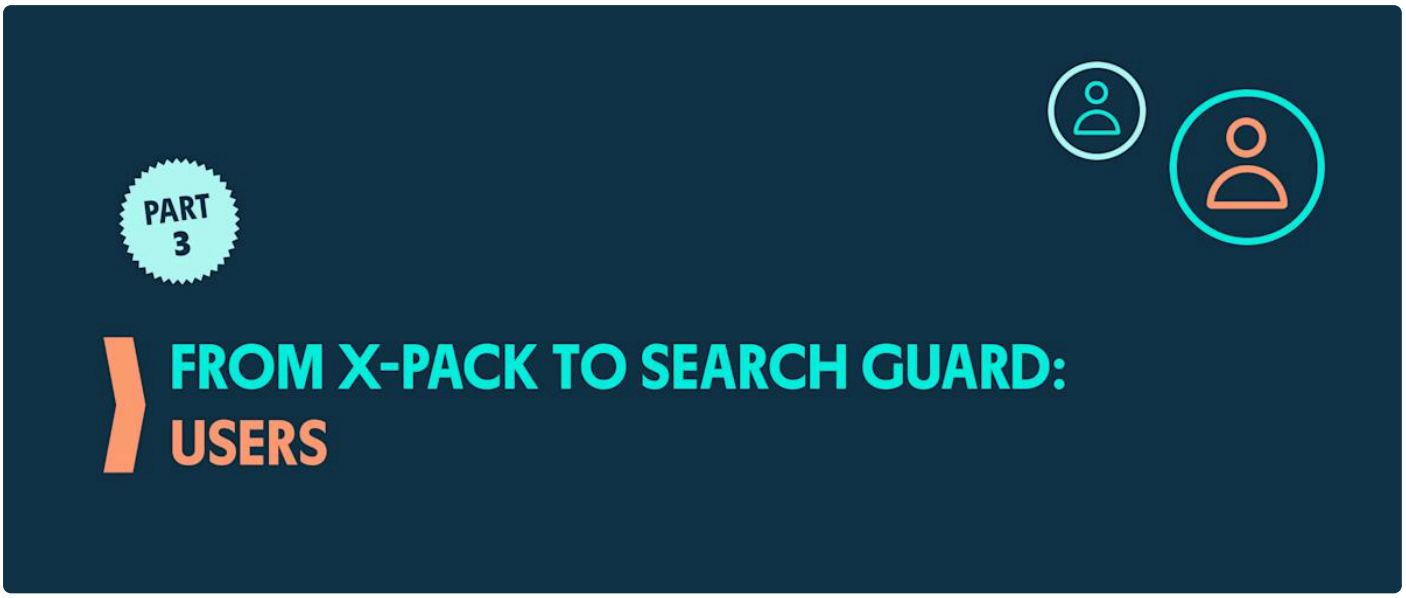
From X-Pack to Search Guard: Part 3 - Users

Jochen Kressin · 2025-07-08



In the journey of migrating from X-Pack Security to Search Guard, one of the most critical steps is transferring your users and their credentials. In this third part of our migration series, we'll focus specifically on how to migrate users from Elasticsearch X-Pack's native realm to the Search Guard internal user database.

Understanding the Differences

X-Pack Security and Search Guard handle users in similar but distinct ways. Let's compare them:

X-Pack Security

- Uses "realms" for authentication, with the native realm being the default
- Stores users in the `.security` index
- Uses the Elasticsearch API for user management

Search Guard

- Uses the internal user database as one of many authentication methods
- Stores users in the `.searchguard` index
- Manages users via `sgctl`, REST API, or the Kibana Configuration GUI

Prerequisites for Migration

Before starting, ensure you have:

1. A running Elasticsearch cluster with X-Pack Security enabled
2. Search Guard installed on a separate Elasticsearch cluster

3. Admin access to both systems
4. The Search Guard sgctl command-line tool installed

Step 1: Extract Users from X-Pack Security

First, we need to retrieve all users from X-Pack Security. This can be done in two ways:

Get User API

Use the Get User API to retrieve the user configured in the native realm. You need at least `manage_security privileges` to access his API.

```
curl -XGET -u elastic:password "https://elasticsearch:9200/_security/user?pretty"
```

This command will return a JSON document containing all users in the X-Pack native realm, including their roles.

The output will look something like this:

```
{
  "elastic" : {
    "username" : "elastic",
    "roles" : [ "superuser" ],
    "full_name" : "Elastic Administrator",
    "email" : "admin@example.com",
    "metadata" : { },
    "enabled" : true
  },
  "kibana_system" : {
    "username" : "kibana_system",
    "roles" : [ "kibana_system" ],
    "full_name" : "Kibana System",
    "email" : null,
    "metadata" : { },
    "enabled" : true
  },
  "john_doe" : {
    "username" : "john_doe",
    "roles" : [ "reader", "reporting_user" ],
    "full_name" : "John Doe",
    "email" : "john@example.com",
    "metadata" : { "department" : "IT" },
    "enabled" : true
  }
}
```

This approach has the disadvantage that no password hash is included. If you plan to assign new passwords to the migrated users in Search Guard, this is not relevant. But if you like to keep the passwords, the next option is more suitable.

Query the .security index (Recommended)

As mentioned above, X-Pack stores users in an index called `.security`. We can query this index directly. This time the password hash is included and can be used to migrate the users with the same password they had in X-Pack. That is what we use in this blog post. To access this index, use the Elasticsearch root user, oftentimes called `elastic`:

```
$ curl -Ss -u "elastic:<password>" https://elasticsearch_node:9200/.security/_search?
pretty&size=10000&q=type:user
```

The output should look something like that:

```
"hits" : [
      {
        "_index" : ".security",
        "_type" : "_doc",
        "_id" : "user-john_doe",
        "_score" : 1.4816045,
        "_source" : {
          "username" : "john_doe",
          "password" : "$2a$12$QzOf63.lc/QaesXkvmk6DOrmmaL001QqMJ403CoDZPgPjLnag/PQC",
          "roles" : [
            "admin",
            "other_role1"
          ],
          "full_name" : "John Doe",
          "email" : "john.doe@example.com",
          "metadata" : {
            "custom_attribute" : "xyz"
          },
          "enabled" : true,
          "type" : "user"
        }
      },

  ...
```

The fields of interest are:

- **_source.username:** The username of the user to login with
- **_source.password:** The hashed password (more on this later)
- **_source.roles:** The roles the user has
- **_source.metadata:** Optional metadata associated with the user

To make this less cluttering you can filter the search request with a tool called jq. This will extract the relevant fields and transform the JSON to a format that is directly consumable by Search Guard:

```
$ curl -Ss -u "elastic:<password>" https://elasticsearch_node:9200/.security-8/_search?
pretty&size=10000&q=type:user | jq '.hits.hits[]._source | {(.username): {hash: .password,
search_guard_roles: .roles, attributes: .metadata}}'
```

Output:

```
{
  "john_doe": {
    "hash": "$2a$12$QzOf63.lc/QaesXkvmk6DOrmmaL001QqMJ403CoDZPgPjLnag/PQC",
    "opendistro_security_roles": [
      "admin",
      "other_role1"
    ],
    "attributes": {
      "custom_attribute" : "xyz"
    }
  }
},
...
```

After checking that the output of our command looks right, we save it in a file called `sg_internal_users.json` which we can upload to our Search Guard secured cluster.

```
$ curl -Ss -u "elastic:<password>" https://elasticsearch_node:9200/.security-8/_search?
pretty&size=10000&q=type:user | jq '.hits.hits[]._source | {(.username): {hash: .password,
search_guard_roles: .roles, attributes: .metadata}}' > sg_internal_users.json
```

Step 2: Mapping X-Pack Roles to Search Guard Roles

When migrating users, this usually also means migrating roles and permissions. In the last article of this series, we created Search Guard roles for all X-Pack roles that are being used in our cluster. As long as the names of the roles are the same, there is nothing more to do.

If you have chosen to give the Search Guard roles a different name, you need to change them in the generated `sg_internal_users.json` file. A simple search & replace will do.

Step 3: Connecting to our Search Guard secured cluster

We have already used the Search Guard Control Tool in the last part of our article to connect to Search Guard. If you have done so already, there is nothing to do since sgtcl will persist the connection settings. If you have not connected to Search Guard with sgctl before, please refer to Part 2 of this article series.

Step 4: Upload the users configuration

The next step is the easiest. The only thing left to do is to uploade our user configuration which we created earlier to Saerch Guard. This can be done with `sgctl` like this:

```
./sgctl.sh update-config /path/to/sg_internal_users.json
```

Which should create an output like this:

```
Successfully connected to cluster mycluster (localhost) as user
CN=kirk,OU=client,O=client,L=test,C=de
Configuration has been updated
```

Conclusion

Migrating users from X-Pack Security to Search Guard is a straightforward process, though it does require some planning. By following the steps outlined in this guide, you can successfully transfer your user base while minimizing disruption to your operations.

With your users now migrated to Search Guard, you can take advantage of the enhanced security features and flexibility that Search Guard offers. In the next part of our series, we'll explore migrating Document- and Field-level security.

Articles in this series

- From X-Pack to Search Guard: Part 1 - TLS Setup
- From X-Pack to Search Guard: Part 2 - Roles and Permissions
- From X-Pack to Search Guard: Part 3 - Users
- From X-Pack to Search Guard: Part 4 - Document- and Field-Level Security - Coming soon!

This article was published on the Search Guard blog: https://search-guard.com/blog/from-x-pack-to-search-guard-part-3-users/