

Using SAML for Kibana Single Sign On

Jochen Kressin · 2018-08-16



Search Guard has full support for SAML. That means you can use your favorite identity provider like Keycloak, Okta or Auth0 to implement easy Kibana Single Sign-On. In this post, we walk you through the steps to set up SAML authentication with Okta.

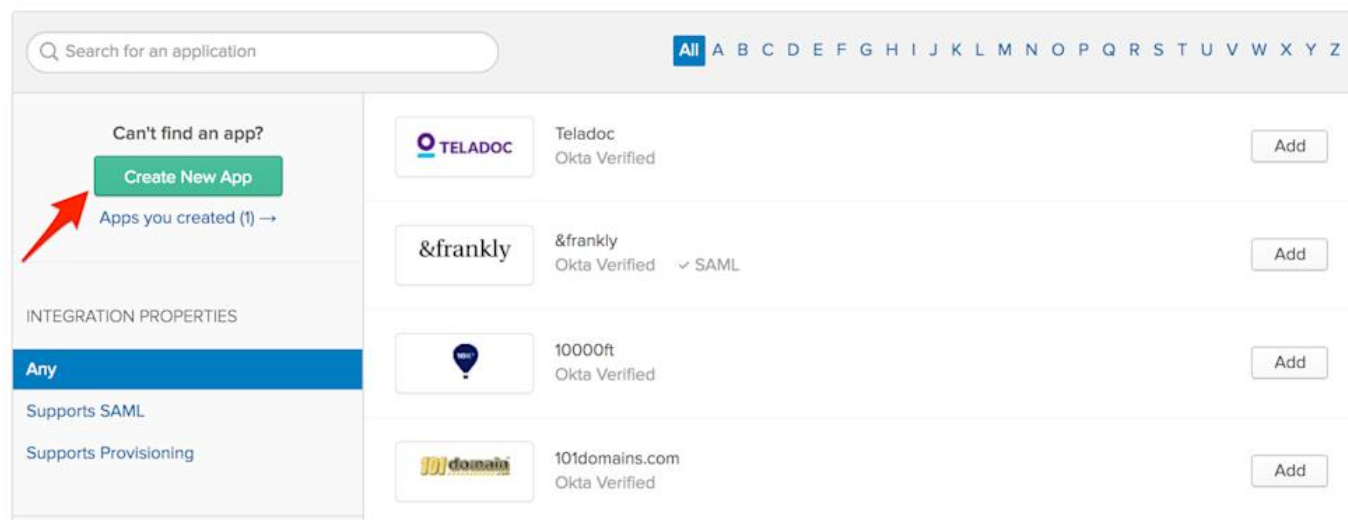
Prerequisites

To use SAML you need to install Search Guard v23 or above and the Search Guard Kibana plugin version 14 or above. In this article, we will use Elasticsearch 6.3.2, so we install

- Search Guard 6.3.2-23.0
- Kibana Plugin 6.3.2-14

Setting up a SAML application in Okta

To use Okta as our identity provider for Kibana we need to first set up a new application with SAML support. After logging in to Okta, click on *Application* → *Add Application* → *Create new application*.

[← Back to Applications](#)[+ Add Application](#)

Search for an application

Can't find an app?
Create New App
Apps you created (1) →

INTEGRATION PROPERTIES

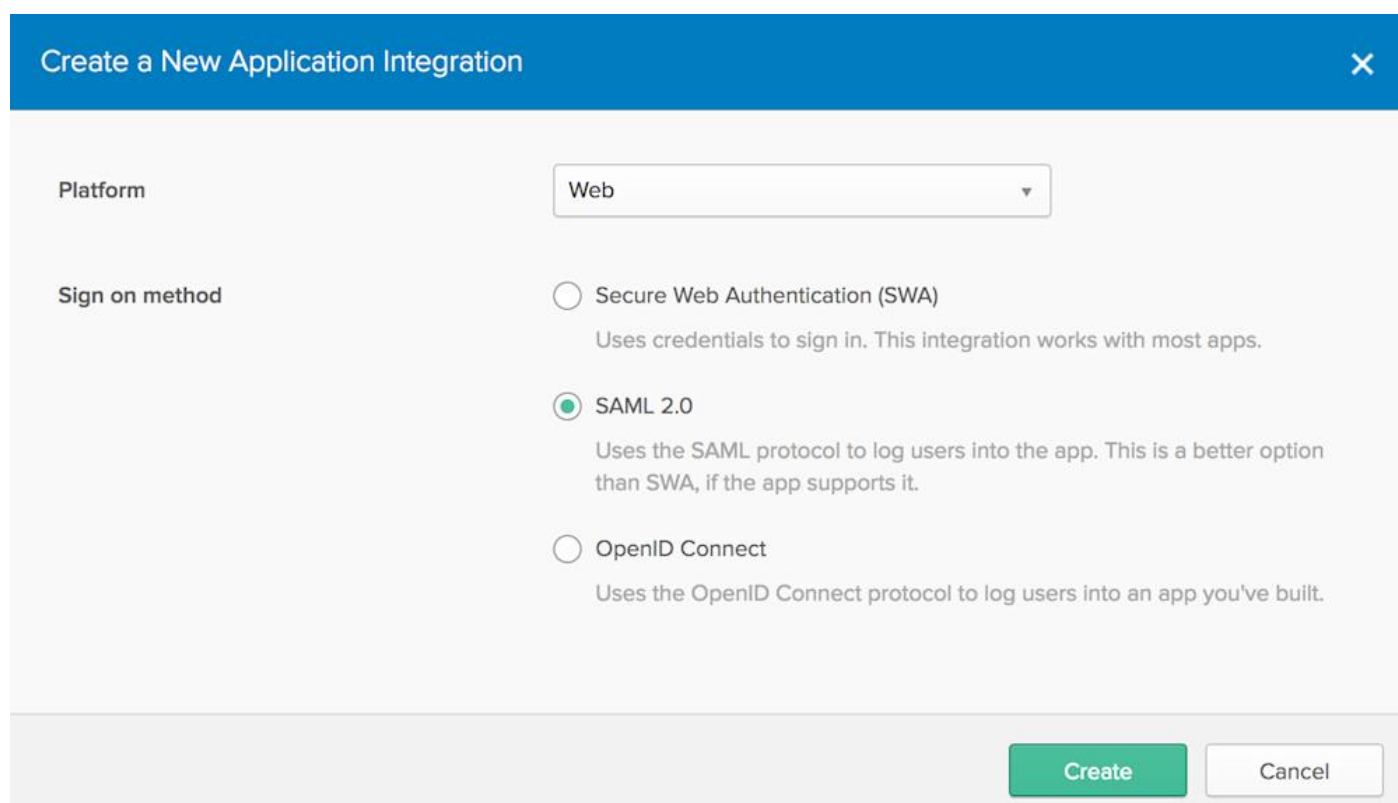
Any

Supports SAML

Supports Provisioning

Application	Okta Verified	Authentication Method	Action
TELADOC	Teladoc	Okta Verified	Add
&frankly	&frankly	Okta Verified SAML	Add
10000ft	10000ft	Okta Verified	Add
101domains.com	101domains.com	Okta Verified	Add

On the next screen, we choose "Web app" as type and "SAML 2.0" as the authentication method.



Create a New Application Integration

Platform: Web

Sign on method:

- ☐ Secure Web Authentication (SWA)
Uses credentials to sign in. This integration works with most apps.
- ☒ SAML 2.0
Uses the SAML protocol to log users into the app. This is a better option than SWA, if the app supports it.
- ☐ OpenID Connect
Uses the OpenID Connect protocol to log users into an app you've built.

Create Cancel

After we have chosen a name for our new application and clicking next, we are taken to the SAML settings screen. The two crucial settings here are the *Single sign on URL* and the *Audience URI (SP Entity ID)*

The *Single sign on URL* specifies the URI where Okta should redirect the request after successful login. This endpoint in Kibana is provided by the Search Guard Kibana plugin, and has the format:

```
https://<kibana_base_url>:<kibana_port>/searchguard/saml/acs
```

Note that Okta requires that the redirect URI be https. Unencrypted HTTP will not work. We will configure Kibana to use HTTPS instead of HTTP in later steps. The *Audience URI (SP Entity ID)* is the identifier for the service provider. You can choose any name here, for this demo we use *kibana-saml-demo*. We need this name later on when we configure SAML support in Search Guard.

GENERAL

Single sign on URL ?

https://kibana.example.com:5601/searchguard/saml/acs

☒ Use this for Recipient URL and Destination URL

☐ Allow this app to request other SSO URLs

Audience URI (SP Entity ID) ?

kibana-saml-demo

Default RelayState ?

If no value is set, a blank RelayState is sent

Name ID format ?

Unspecified ▼

Application username ?

Okta username ▼

Show Advanced Settings

Adding roles to the SAML response

Since we want to use the Okta groups of a user to assign Search Guard roles, we need to make sure that these groups are added to the SAML response when the user logs in. In Okta this is achieved by adding Group Attribute Statements. The *Name* field defines the name of the SAML assertion that will hold the groups of the user. Since we want to add all groups of the user to the SAML assertion, we will use a regular expression as a filter that matches all groups:

GROUP ATTRIBUTE STATEMENTS (OPTIONAL)

Name	Name format (optional)	Filter	
Role	Unspecified ▼	Regex ▼	.*
Add Another			

Metadata endpoint

After finishing the setup wizard, we're taken to the *Settings* screen. Here we get access to the *SAML metadata*. This metadata makes it possible to auto-configure the client application and contains information about endpoints, formats, encryption and ciphers.

We click on *Identity Provider metadata* and save the XML file in the config directory of Elasticsearch. We name the file *okta.xml*:

SIGN ON METHODS

The sign-on method determines how a user signs into and manages their credentials for an application. Some sign-on methods require additional configuration in the 3rd party application.

SAML 2.0

Default Relay State



SAML 2.0 is not configured until you complete the setup instructions.

[View Setup Instructions](#)

[Identity Provider metadata](#) is available if this application supports dynamic configuration.

Adding users and groups

Now it's time to add a user and the groups of the user. We will start with the groups. Our goal is to create a user with read-only access to all indices and access to Kibana. Since we use the Search Guard demo configuration, we want the user to have the following backend roles:

- kibanauser
- read_all

These backend roles are mapped to the corresponding Search Guard roles in the demo configuration. So we click on *Directory* → *Groups* → *Add Group* to create the *kibanauser* and *readall_* group.

Add Group

Add groups so you can quickly perform actions across large sets of people.

Name	readall
Group Description	Enter a description for this group...

Add GroupCancel

We then create a user by clicking on *Directory* → *People* → *Add Person*. We assign the newly created roles kibanaserver and readall.

Add Person

First name

John

Last name

Doe

Username

john.doe@example.com

Primary email

john.doe@example.com

Secondary email (optional)

Groups (optional)



kibanauser ×



readall ×

Password ?

Set by admin ▼

....

☐

User must change password on first login

Save

Save and Add Another

Cancel

Finally, we need to assign the user jdoe to the Kibana SAML Demo application:

General

Sign On

Import

Assignments

Assign ▾

⚙️ Convert Assignments

🔍 Search...

People ▾

FILTERS	Person	Type	
People	 John Doe jdoe@example.com	Individual	 
Groups			

Configure SAML for Elasticsearch

Now we have everything in place for configuring SAML for both Elasticsearch and Kibana. Since Search Guard can read and make use of the published SAML metadata, the actual configuration is pretty straight-forward.

Our goal is to use SAML for Kibana, but still being able to access Elasticsearch API directly. So we configure two authentication domains, one leveraging Basic Authentication and the internal user database, and one for SAML support:

```
authc:
  basic_internal_auth_domain:
    enabled: true
    order: 0
    http_authenticator:
      type: basic
      challenge: false
    authentication_backend:
      type: intern
  saml_auth_domain_okta:
    http_enabled: true
    transport_enabled: true
    order: 1
    http_authenticator:
      type: 'saml'
      challenge: true
      config:
        idp:
          metadata_file: okta.xml
          entity_id: http://www.okta.com/exkflvnptwlnzRkp40h7
        sp:
          entity_id: kibana-saml-demo
        kibana_url: https://kibana.example.com:5601/
        roles_key: Role
        exchange_key: 'peuvvg0LrjzuhXfRUwzU...'
```

idp.metadata_file

This is name of the metadata XML file we downloaded from Okta. The file must be placed in the *config* directory of Elasticsearch. The *metadata_file* path must be relative to the *config* directory.

idp.entity_id

This is the unique ID of your identity provider. You can find this ID for example in the SAML metadata:

Optional

- 1 Provide the following IDP metadata to your SP provider.

```
<?xml version='1.0' encoding='UTF-8'?><md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
entityID="http://www.okta.com/exkfxs0k5yBYyllp0h7"><xsi:type="IDPSSODescriptor WantAuthnRequestsSigned="false"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:oasis:names:tc:SAML:2.0:protocol"><md:KeyDescriptor use="signing"><ds:KeyInfo
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"><ds:X509Data>
<ds:X509Certificate>MIIDpDCCAoygAwIBAgI GAWPALuxAMAOGCSqGSIb3DQEBCwUAMIGSMQswCQYDVQQGEwJVUzETMBEG
A1UECAwKQ2FsaWZcm5pYTEWMBQGAIUEBwwNU2FuIEZyYW5jaXNjbzENMAsGA1UECgwET2t0YTEU
MBIGA1UECwwLU1NPUHJvdmlkZXIxExARBgNVBAMMCmRldiOONjlnZjExHDAABgkqhkiG9w0BCQEW
DWluZm9Ab2t0YS5jb2UwHhcNMTEgNWJlAYMTewNzEwWhcNMjgwNjAyMTewOWEwVjCBKgELMAKGA1UE
BhMCMVMVMAcxERBgNVBAgMcKhnbGlmb3JuaWEwExFAUBGxNVBAcMDVNhbibiBGcmFuZy21Zy28xDALBgNV
BAoMBE9rdGExFDASBgNVBASMCINTT1Byb3ZpZGVyYMRMwEQYDVQQDDApkZXZYNTNDYyMzYxMRwwGgYJ
KoZihvcNAQAkBfBg1pbmZvQG9rdGEUy29tMIIBljANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
btAvTUhm0hdDOTXHhHPGbPd+g4IRuYaaLMRZF2BJs5S5CrF5ApbQ847kzs05P4oP3usVBnc2g94
Ztkk236DXphQUSvKHzbSpdTrryZjFPgo8xBvILC6KA8zKYeZ01k9ytIEeS242w1682+8Y1vb
gkc2qBjocvSerbkZdaEIziU4zfX0SFuPghCRxD7gU0bn9ludZukN9XOWDjkQJZIkkzDBsAku5u7
dCc203KUUI1LZuBpm3DzXYuFpqfw7HbDgsllrt6TDpkotvP3/b8LIjm2K/OxaFG4xYg2QVt4r6r
8H9nGm4kchAbraaoHtULlzCOHASrHqkgOfwiIDAQABMAOGCSqGSIb3DQEBCwUAA4IBAQBWXCs
7xvvmyVLJ4mdSYgzLTBiZQORUKdyQl5CTQYasrg/R/7BxvYM2cU5zwJsXWzJLLkkyBm21jw8ZBIA
GYQMVEyoM7L63khokGozQt7dHuKM8OflixS35oLBxje5vJhuUAcboDIowje65yd4IKbGLig+vb
Fc/d+MEq9iR7OW0+mOY+XuMoxAwlpXZ49GR0je6zs2q7qVVtuLuRg6vMgFUncp0xgryBZPNWxOBF
MTE9neBuOHwhya4KFdcCYvirAun+cXOdChFbFe00vVrF368dIDK0ICKjpp0v/NDgunwlbk7MSVTOncc
ednbluKE=Kf5A45F5FAD3z3z1AWZ-</ds:X509Certificate></ds:KeyInfo></md:KeyDescriptor>
```

sp.entity_id

This is the service provider entity ID. We configured this name under *Audience URI (SP Entity ID)* when we set up the SAML application in Okta.

kibana_url

This is the base URL of your Kibana installation.

roles_key

The attribute name under which the roles can be found in the SAML response. We have used the name *Role* when we configured the group attribute statements during setup.

exchange_key

SAML, unlike other protocols like JWT or Basic Authentication, is not meant to be used for exchanging user credentials with each request. Therefore Search Guard trades the heavy-weight SAML response for a light-weight JSON web token. This token is signed with an exchange key that you can choose freely. The key should be at least 32 characters long.

Configure SAML for Kibana

Since most of the heavy lifting is done by Search Guard, the Kibana configuration is straightforward. We need to configure SAML as the authentication type, and whitelist the redirect and logout URL:

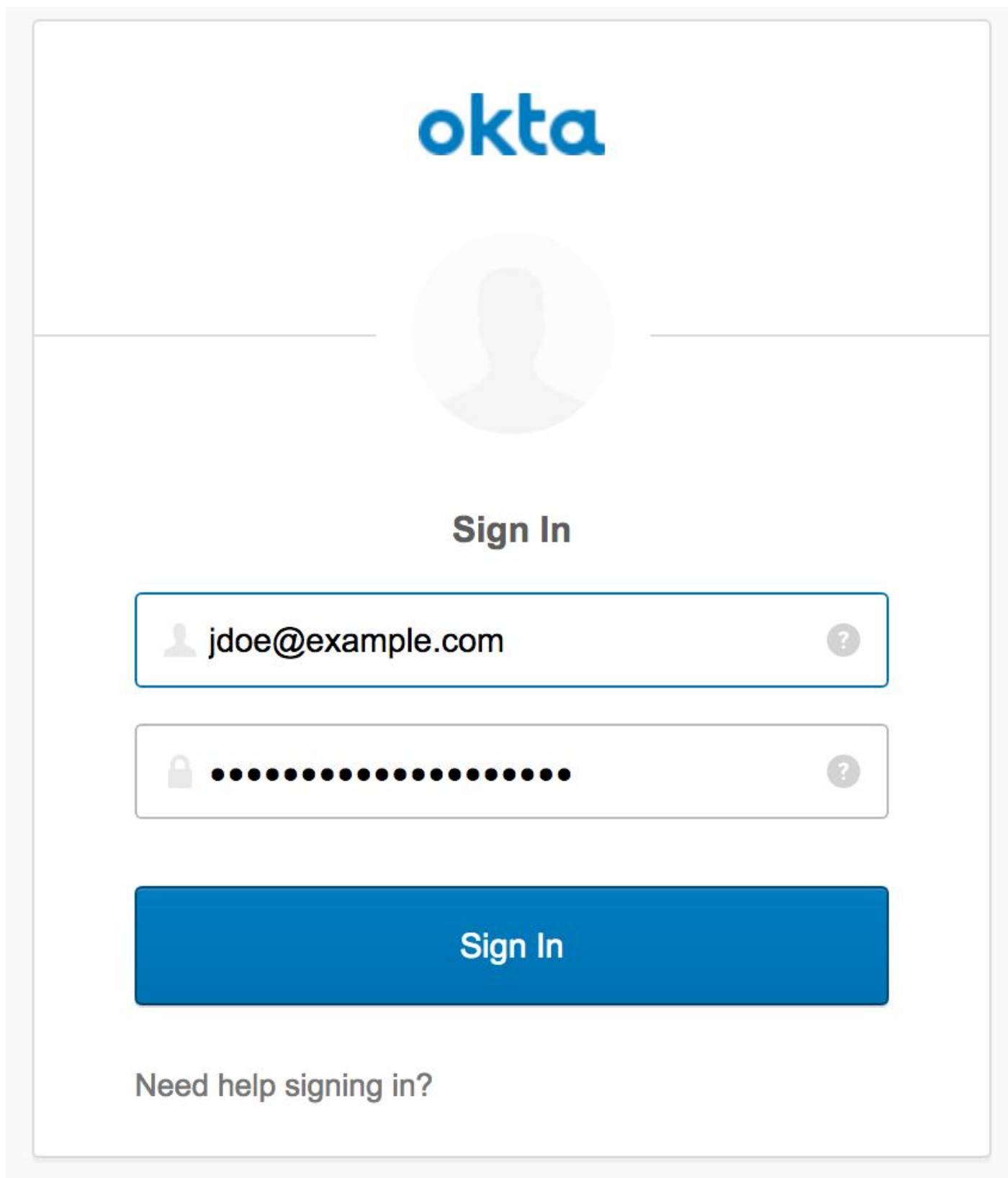
```
searchguard.auth.type: "saml"
server.xsrf.whitelist: ["/searchguard/saml/acs", "/searchguard/saml/logout"]
```

Since Okta requires the service provider (in our case Kibana) to use HTTPS, we configure the SSL certificate and key as well:

```
server.ssl.enabled: true
server.ssl.certificate: ./config/kibana.crtfull.pem
server.ssl.key: ./config/kibana.key.pem
```

Trying it out

Now we're ready to try our installation. We access Kibana via <https://kibana.example.com:5601> and get redirected to the Okta login page:

A mockup of an Okta sign-in page. At the top is the 'okta' logo in blue. Below it is a large, light gray circular placeholder for a user profile picture. Underneath the picture is the text 'Sign In' in a bold, dark gray font. There are two input fields: the first is for an email address, containing 'jdoe@example.com' with a small user icon on the left and a help icon on the right; the second is for a password, represented by a series of black dots with a lock icon on the left and a help icon on the right. Below these fields is a large blue button with the text 'Sign In' in white. At the bottom of the form is the text 'Need help signing in?' in a dark gray font.

okta

Sign In

jdoe@example.com

●●●●●●●●●●●●●●●●

Sign In

Need help signing in?

After we log in, we get redirected to Kibana and can start using it.

Summary

In this article, we demonstrated how to setup SAML authentication for Kibana Single Sign-On. We used [Okta](#) as our identity provider. Since SAML is an industry standard, there is a wide variety of IdPs to

choose from, including [Auth0](#), [Keycloak](#) or [OneLogin](#).

Where to go next

- Read the official [SAML configuration documentation](#)
- If you have trouble configuring SAML, refer to our [SAML troubleshooting guide](#)
- If you have any question, ask them on our [Search Guard Forum](#)

Image: shutterstock / [jirsak](#)

This article was published on the Search Guard blog: <https://search-guard.com/blog/kibana-elasticsearch-saml/>
Search Guard is Security and Alerting for Elasticsearch and Kibana — search-guard.com · [Start a free trial](#)