SIEMENS

Single Sign On (SSO) with Polarion 19.3

POL007 - 19.3

Contents

Configure single sign-on (SSO)

Introduction 1-1
Supported browsers and versions 1-1

SAML SSO

SAML authentication procedure flow 2-1
Prerequisites 2-1
Basic Configuration 2-2
- Polarion Configuration 2-2
- IdP configuration 2-3

Optional configuration 2-4
Other notes 2-7

Kerberos SSO

Authentication procedure flow 3-1
Prerequisites - configuring the domain 3-2
Polarion configuration 3-4
Configuring client browsers 3-5
Authentication fallback for external users using Kerberos 3-9

Teamcenter SSO

Authentication procedure flow 4-1
Prerequisites - TcSS configuration 4-2
Polarion configuration 4-2

Other SSO considerations

Subversion access 5-1
Direct access to subversion 5-2
Features requiring special configuration in an SSO setup 5-4
Subversion repository access during Polarion update 5-4
Internal log-on pop-up behavior 5-4

Troubleshooting

SSL 6-1
Enable logging of SAML response 6-1
Enable international file encoding support 6-2
Kerberos hints 6-3

1. Configure single sign-on (SSO)

Introduction

This section describes SSO authentication methods and configurations for Polarion. Polarion supports authentication using *Security assertion markup language 2.0* (SAML), **Kerberos** tokens and **Teamcenter security services** in addition to the existing authentication methods.

Supported browsers and versions

Polarion ALM is web-based software. For the best experience, always use a supported browser. Polarion may work with other browsers, but only those listed here have been tested and certified as supported. Polarion displays a warning message to users who log in using an unsupported browser.

Browser	Version(s)	Notes
Google Chrome	Latest	
Mozilla Firefox	68 or newer	Version 71 for users hoping to tap into the very latest features and improvements. Organizations or individuals looking for longer-term support with less frequent browser updates should use Version 68 ESR
Microsoft Edge	Latest	
Microsoft Internet Explorer	11	Please be sure to review the additional notes section, below, for important notes about this browser.

Additional notes for Microsoft Internet Explorer

- If running Internet Explorer on Windows Server 2012 (for example, for testing):
  - If prompted by the browser, you will need to add **about:blank** as a trusted server.
  - You may need to add the local Polarion server and any remote Polarion servers as trusted servers.
  - You need to enable JavaScript for the Polarion server.

> Note:
>
> The above points do not apply when accessing Polarion running on Windows Server 2012 from other clients.

- Users of Internet Explorer should specify their Polarion server in the Local Intranet security zone while ensuring that Compatibility View is not used.

- There can be performance issues on client computers running Internet Explorer due to browser memory requirements. Occasional restarts of the browser are recommended.

2. SAML SSO

SAML authentication procedure flow

Polarion supports the standard Security Assertion Markup Language (SAML 2.0).

The authentication procedure flow is as follows:

1. The client machine connects to Polarion and if a user is not authenticated, Polarion redirects them to the Identity Provider (IdP).

2. The user then needs to be authenticated in the IdP to obtain a SAML response.

3. The IdP will then redirect them back to Polarion.

4. The browser provides this SAML response when negotiating the Polarion authentication window.

5. The Polarion server then validates the SAML response based on SAML assertions.

6. If Polarion successfully validates all the assertions, it authenticates the user and logs them in.

For more info and the glossary see the following links:

- **SAML Technical Overview**
- **SAML Glossary**

Prerequisites

The following prerequisites are required in order to configure Polarion to authenticate using SAML:

1. A standard IdP needs to be installed on the network and connected to the company's directory service or another source. (This Guide will use the Microsoft ADFS as an example). IdP needs to be installed and available to the client machines on, for example, *https://saml-server.yourdomain.com/adfs/ls*.

2. The Polarion and IdP machines need to be configured for the HTTPS protocol. See **Administrator's Guide→ Configure SSL Support** in Polarion's Help for details.

> Warning:
>
> OCBBlockCipher encoding is not supported.

Basic Configuration

Polarion Configuration

Before configuring Polarion to use SAML SSO you need to enable Auto-Create in Polarion.

1. Go to **Global Administration** in Polarion. First click ⚙ on the top left.
   (Then **Administration** → **Global Administration** → **User Management**.)

2. Click **Auto-create** and check **Enable Auto-create**.

3. Set the **Global Rules** given to newly created users.

> Note:
>
> When SAML SSO is turned ON, you will only be able to login via the IdP. Therefore you should either create a User with admin rights that has the same User ID that the IdP will use, or add the admin Role to the Global roles for auto-create, and then after logging in via SAML SSO for the first time change the Global roles.

> Note:
>
> Logon via SAML SSO changes the passwords for these users, so it is advisable to leave at least one user with admin rights that will not be used by SAML SSO.
>
> (Just in case you turn SAML SSO off in the future.)

Add Properties to the polarion.properties file:

1. Configuring Polarion to use SAML authentication requires setting the following property that enables the SAML logon:
   com.siemens.polarion.security.auth.method=saml located at:
   Windows: C:/Polarion/polarion/configuration/polarion.properties
   Linux: /opt/polarion/etc/polarion.properties

2. And then depending if you are using the metadata:

   a. If you want to use IdP Metadata, set these properties:
   com.siemens.polarion.security.saml.identityProviderMetadataUrl=
   <Url to the IdP Metadata>, for example:
   "*<file:///C:/Path/FederationMetadata.xml>*"
   Or if you have the metadata available on the remote machine:
   "*<https://saml-server.yourdomain.com/FederationMetadata/2007-06/FederationMetadata.xml>*"

The file protocol is recommended for metadata configuration.
(You don't need to worry about the certificate. It should be located within the Metadata.)

> Caution:
>
> You can always overwrite the settings in Metadata by manually specifying other properties. (See the list of properties in the next step.)

b. If you do not have access to the IdP Metadata, you can specify the SSO Service URL manually by using the following property: **com.siemens.polarion.security.saml.ssoServiceUrl=***<Your IdP Service URL> , e.g "<https://saml-server.yourdomain.com/adfs/ls">*.
If you do, you must also provide the certificate file needed for SAML assertion validation that needs to be exported from the IdP. By default, you should export the certificate to the **%POLARION_HOME%\polarion\configuration** folder. You can specify a different path to a certificate using the **com.siemens.polarion.security.saml.certificatePath** property.
The other properties have these default values and should work implicitly, but you can also specify them if you need to:
com.siemens.polarion.security.saml.relyingPartyIdentifier=*<Polarion base.url>*
com.siemens.polarion.security.saml.assertionConsumerServiceUrl=*<Polarion base.url>/polarion/>*
com.siemens.polarion.security.saml.issuer=*<IdentityProviderUrl base.url>*
com.siemens.polarion.security.saml.forceAuthn=*false*
com.siemens.polarion.security.saml.certificatePath=*<By default a certificate in the configuration folder is used>*.

3. After setting the properties, restart the Polarion Service, and the SAML SSO should be working.

IdP configuration

The Identity provider must be properly configured to communicate with Polarion correctly.

- To do this, you need to configure the Relying Party Identifier for Polarion in the IdP registry, so it points to Polarion's server base url, for example, `https://polarion.yourdomain.com`.

> Caution:
>
> In most cases do **NOT** include `/polarion` at the end, but it's required for the Okta **Single Sign on URL** field.

- If the IdP configuration wizard supports auto-configuration using SAML metadata, an administrator can point to the Polarion endpoint that exposes this metadata. They are available at the following url: *https://polarion.yourdomain.com/polarion/saml2/sp/metadata*
Metadata will be accessible only when Polarion is run with the following property added to the **polarion.properties** file.
com.siemens.polarion.security.auth.method=saml

- The IdP must be configured to send the ID in the **Name ID** attribute. This attribute is then used by Polarion as the User ID.

ADFS Example:

1. In ADFS, go to the relying party trusts folder and add a new relying party trust.

2. A wizard opens and takes you through the configuration. You can use the Polarion metadata mentioned earlier, or a manual configuration, using the *https://polarion.yourdomain.com* as the **Relying SAML 2.0 SSO service URL** and the **Relying party trust identifier**.

3. After the set up is complete, you should be prompted to add a **Claim rule**. If not, you can always right click on your **Relying Party Trust** and edit the claim rules from there.

4. When adding the **Claim rule**, choose **Send LDAP Attributes as Claims** and configure the mapping to send an LDAP attribute as a **Name ID** attribute. For example, **SAM-Account-Name → Name ID**

This is the minimum that needs to be configured on the IdP side for SAML SSO to work with Polarion. For more options, see the *Optional configuration* section.

Optional configuration

The properties configured in the Basic configuration are sufficient for Polarion to work with SAML SSO, but there are additional optional features that Polarion Supports.

Single logout

To enable Single logout via the IdP, you need to configure the following properties:

1. **com.siemens.polarion.security.saml.sloServiceUrl=** a link to your IdP logout service, for example, *<"https://saml-server.yourdomain.com/adfs/ls/?wa=wsignout1.0">*

2. **com.siemens.polarion.security.ssoHomePageUrl=**a link to your IdP Home Page, for example, *<"https://saml-server.yourdomain.com/adfs/ls/IdpInitiatedSignOn.aspx">*

The **sloServiceUrl** property is the URL to that Polarion sends the logout response to.

The **HomePageUrl** is just a link that appears when you log out from Polarion, so that you can continue to your IdP where you can logout from SSO. (Polarion only supports IdP Initiated Single Log-out.)

You also need to configure your IdP accordingly. The link you need to provide to your IdP is:

https://polarion.yourdomain.com/polarion/login/logout

ADFS Example- To set up Single Logout in ADFS:

1. Open ADFS.
2. Open your Relying Party Trust.
3. Select the Endpoints tab and click **Add SAML...**
4. Select the **SAML Logout** type, and;
    a. Set the Trusted URL to *https://polarion.yourdomain.com/polarion/login/logout*
    b. Set the Response URL to *https://saml-server.yourdomain.com/adfs/ls/?wa=wsignout1.0*
5. Set the following properties in Polarion's **polarion.properties** file:
com.siemens.polarion.security.saml.sloServiceUrl=*https://saml-server.yourdomain.com/adfs/ls/?wa=wsignout1.0*
com.siemens.polarion.security.ssoHomePageUrl=*https://saml-server.yourdomain.com/adfs/ls/IdpInitiatedSignOn.aspx*

Synchronization of attributes

You can configure SAML to send more attributes than just the Name ID. In Polarion, the user attributes that can be automatically updated from the SAML response are Name, Email and Description. You can configure the names of the SAML attributes used to fill these user attributes by using these three properties:

- **com.siemens.polarion.security.saml.userMapping.name=**<name of the SAML attribute, default is "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name">
- **com.siemens.polarion.security.saml.userMapping.email=***<name of the SAML attribute, default is"http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress">*
- **com.siemens.polarion.security.saml.userMapping.description=***<name of SAML attribute, by default it is empty (nothing will be filled in description if this property is not set up)>*

By default the Name and Email attributes are set to the names used by ADFS. However "description" is not set to any SAML attribute by default and you need to specify the attribute to be used there.

> Note:
>
> The attributes will be updated with every Log on to Polarion, but if the received SAML data is empty, there will be no change in the User attributes. (If a user does not have, for example, the e-mail filled in in the IdP, you can fill it in manually in Polarion and it will not be saved as empty upon logon). You can check the main log to see the SAML response if you have trouble with the exact name of the SAML attribute. (See the *Enable logging of SAML response* section for details.)

ADFS example- In ADFS:

1. Open Relying Party Trusts, right click on your application and select **Edit claim rules**.

2. Here you can either edit an existing rule, or add a new one.

3. You will need to add mapping for attributes you want to be sent, for example: "**E-Mail Addresses**" → "**E-Mail Address**"; "**Display-Name**"→ "**Name**"; "**Department**"→ "**Role**", and then save the rule.

4. Since the **UserMapping** properties for **Name** and **Email** in Polarion are set for **ADFS** by default, you don't need to configure them. You need to configure the Description property however, so in our example you will need to add this property into the **polarion.properties** file:
com.siemens.polarion.security.saml.userMapping.description=
http://schemas.microsoft.com/ws/2008/06/identity/claims/role

5. After restarting Polarion and logging in as a **User**, the **Name**, **E-mail** and **Description** will be filled with the "**Display-Name**", "**E-mail Addresses**" and "**Department**" attributes from AD.

You can choose different attributes while setting up the claim rules, and then change the Polarion properties accordingly.

Alternative log on

If you want to use a different **User ID** in Polarion than the one that is sent via SAML (for example you have existing Users in Polarion with different IDs than their Account Names in AD), you can configure sending different IDs in your IdP, if it supports it. You can also use the alternative ID Polarion property. You can map a SAML attribute to this property, and Polarion will use this SAML attribute as the User ID, but only if it is not empty, otherwise it will still use the "**Name ID**" SAML attribute.

com.siemens.polarion.security.saml.userMapping.alternativeId=

<name of the SAML attribute that will be used as the User ID if it is not empty>

> Warning:
>
> The attribute you will use as the alternative Login ID needs to be unique for your users, otherwise users with different IdP Account names, but with the same content in the Alternative ID attribute, will be logged on as the same Polarion User. Additionally, if you want to use E-Signatures in Polarion, it is not recommended to use the Alternative ID function. If there is no other way, you need to configure LDAP correctly with the alternative ID. See the *E-signature* section for details.

Authentication request signing

By default, Polarion does not sign an authentication request when it sends it to the Identity Provider, and the Identity Provider will not demand that the request is signed. However, if your company policy mandates that SAML requests should be signed, it can be done by specifying the following two properties:

```
com.siemens.polarion.security.certificate=path_to_certificate_file
```

For example, `C:\\Path\\to\\Certificate\\certificate.crt`

`com.siemens.polarion.security.privateKey=path_to_private_key_file`

For example, `C:\\Path\\to\\PrivateKey\\privatekey.key`

> Warning:
>
> The Identity Provider will require the certificate file (or its contents), to verify your signature. For more information about mandatory request signing, read your IdP documentation.

Other notes

Permissions to log on

Polarion will always auto-create a user with configured roles when it receives a valid SAML response with a new User ID. If you need to deny some of your SSO users access to Polarion, set the permissions in your IdP. If Polarion receives a invalid or denial response it will not create the new User.

E-Signatures with SAML

Since IdPs do not support logging on in an IFrame, it is not possible to use E-Signatures in Polarion with only SAML authentication configured. To use E-Signatures with SAML SSO you need to configure LDAP E-Signatures. See **User Guide → 15. Working with Documents → Signing Documents → In an SSO Environment** in Polarion's Help for details.

> Warning:
>
> Since E-Signatures validate the signature based on the Polarion User ID and the specified ID used by LDAP, you need to make sure that they match. If you use a different attribute than the Account ID in the SAML set up, you will need to configure LDAP to the same attribute. It is therefore not recommended to use the alternative ID when you want to use E-Signatures.

3. Kerberos SSO

Authentication procedure flow

Polarion supports standard Kerberos v5 authentication using the GSS-SPNEGO mechanism.

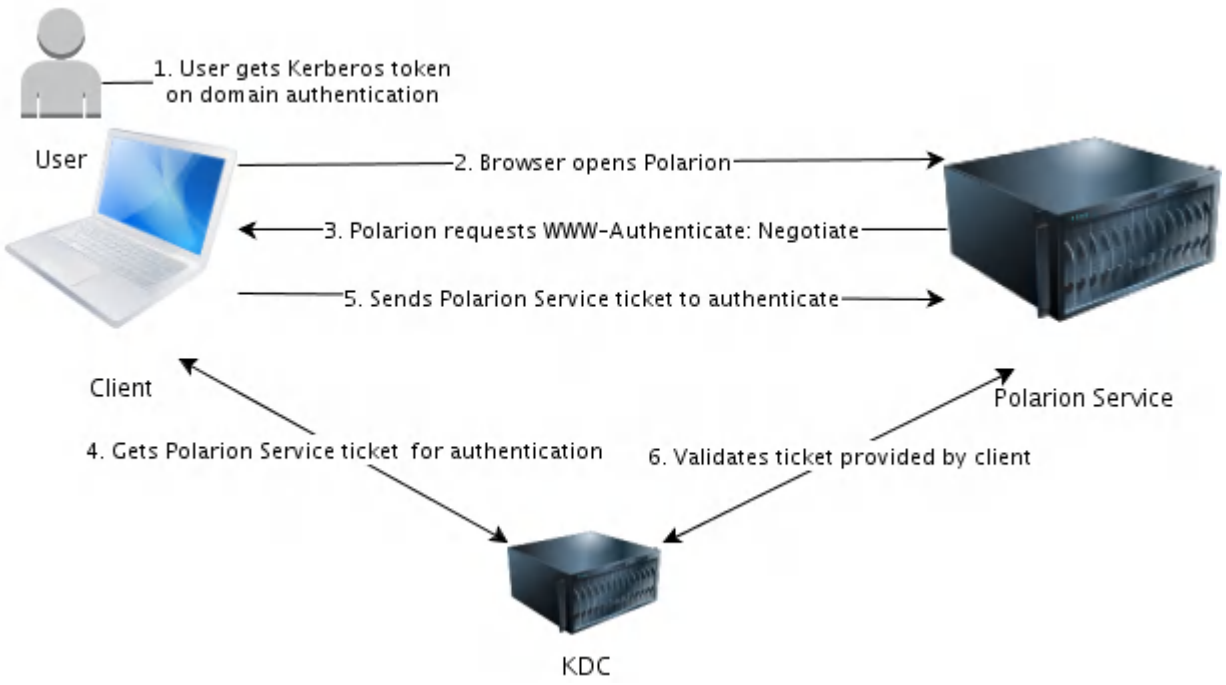
The authentication procedure flow is as follows:

1. The client machine logs into the domain and obtains a Kerberos token.

2. If the browser is configured to use the Kerberos token for authentication, the browser sends this token when negotiating authentication with a site that is a member of the same domain.

3. The Polarion server gets the token from the client and validates it against the domain controller.

4. If the domain controller successfully validates the token, Polarion authenticates the user and logs them in.



Prerequisites - configuring the domain

A set of prerequisites is required in order to configure Polarion to authenticate using Kerberos tokens:

1. All involved machines need to be members of the same domain for example, *yourdomain.com*

    a. Domain server (*kdc.yourdomain.com*) that hosts KDC (Key Distribution Center) and DNS services.

    b. Domain users (for example, user1) that log into client machines.

    c. Polarion service provider (for example, *polarion-server.yourdomain.com*) that represents the server running Polarion.

2. Client and Polarion machines need to have corresponding domain accounts (principals):
   Domain user: *user1@YOURDOMAIN.COM*
   Polarion service: *HTTP/polarion-server@YOURDOMAIN.COM*

3. Client browsers are configured for Kerberos authentication.

4. A keytab file with the password key for the Polarion service principal needs to be generated on the domain server.

    a. On Windows run the following command as an administrator on your domain server.

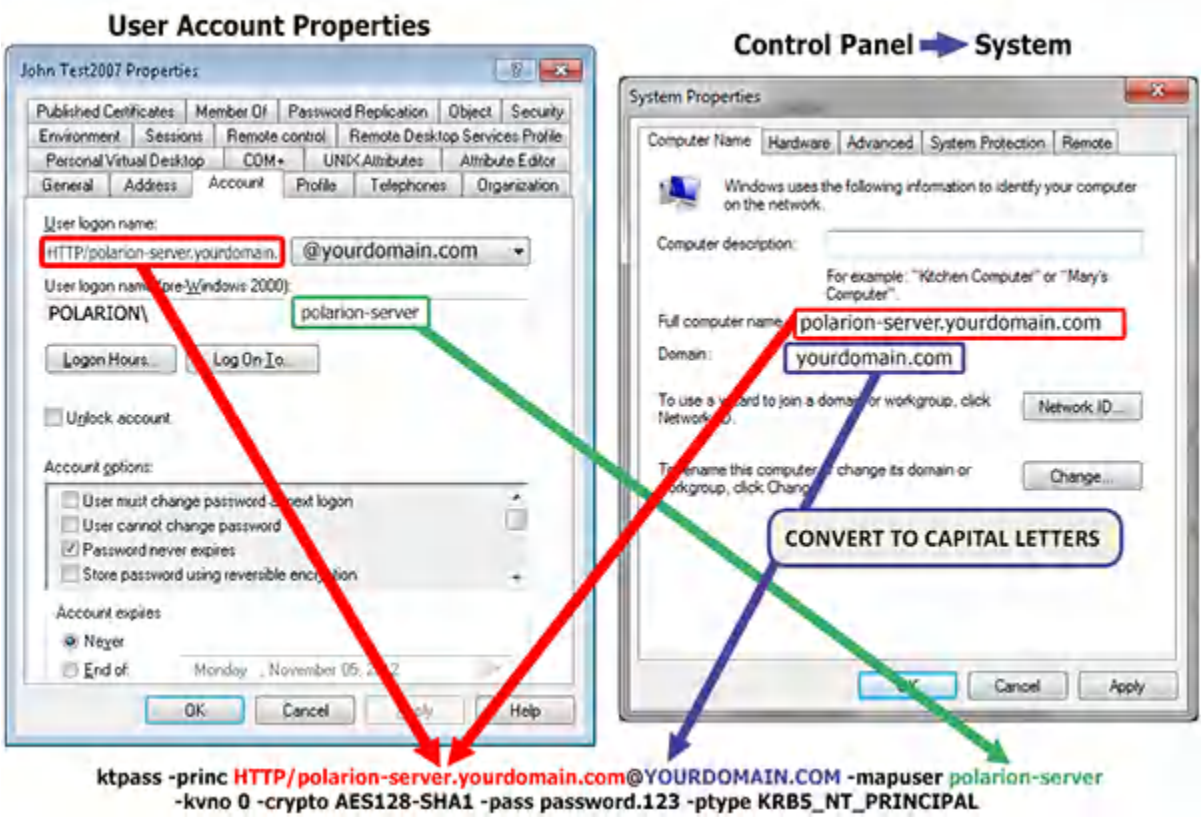
    ```
    ktpass -out polarion.keytab -princ HTTP/polarion-
    server.yourdomain.com@YOURDOMAIN.COM -mapuser polarion-server -kvno
    0 -crypto AES128-SHA1 -pass password.123 -ptype KRB5_NT_PRINCIPAL
    ```

    > Note:
    >
    > You should align the keytab encryption with the domain's user settings.
    >
    > (HTTP must be in the principal name.)

    b. On Linux run the following command as a root user on your Domain server replacing *yourdomain* with your own.

    ```
    ktadd -norandkey -k /tmp/polarion-server.keytab HTTP/polarion-
    server.yourdomain.com@YOURDOMAIN.COM
    ```

See the example to follow and refer to the correct values for each **ktpass** command attribute.



Notes:

- The HTTP service component in the principal name is required, otherwise a Kerberos authentication with the polarion-server will not work.
- The Polarion service provider principal must contain the fully qualified domain name of the machine's (FQDN).
- Polarion's Kerberos SSO authentication requires that principals use AES 128 bit encryption.

1. In the Windows domain, all user accounts must have the following configuration checked: **Properties→ Account→ Account options→ This account supports Kerberos AES 128 bit encryption.**
2. In the Linux realm, the encryption needs to be set properly when adding the following principal: For example:
   addprinc -e aes128-cts -randkey HTTP/polarion-server.yourdomain.com@YOURDOMAIN.COM
3. The DES encryption type is not considered secure, so it is not allowed in the default configuration of many operating systems.

- The generated keytab needs to respect this configuration as well, so make sure to set the **-crypto** option below so that it's in line with the principal configuration.

- Kerberos always passes the **sAMAccountName** attribute as the user's principal name, so this field also represents the **userId** in Polarion. If you need to synchronize additional user attributes from LDAP, please refer to the **Administrator's Guide → Managing Users & Permissions → Configuring Polarion for LDAP User Synchronization** section in Polarion's Help.

Polarion configuration

Polarion configuration consists of a few simple steps:

1. Copy the generated **polarion.keytab** file to the Polarion server under Polarion's **configuration** directory.

2. On the Polarion server, create a **login.conf** file in Polarion's **configuration** directory with the following data:
```
Polarion {
com.sun.security.auth.module.Krb5LoginModule required
doNotPrompt=true
principal="HTTP/polarion-server.yourdomain.com@YOURDOMAIN.COM"
useKeyTab=true
keyTab="C:/Polarion/polarion/configuration/polarion-server.keytab"
};
```

3. Configure Polarion to use SPNEGO authentication and set the following properties in: **C:/Polarion/polarion/configuration/polarion.properties**

```
com.siemens.polarion.security.auth.method=spnego

java.security.krb5.realm=YOURDOMAIN.COM

java.security.krb5.kdc=kdc.yourdomain.com

java.security.auth.login.config=C:/Polarion/polarion/configuration/
login.conf
```

Once you have successfully configured the domain, the domain server, the domain client and the Polarion server, users will not see the Polarion login page, but will be immediately authenticated using the provided Kerberos token and logged into Polarion as the same user with same login ID.

Configuring client browsers

All currently **supported browsers** can be configured to use Kerberos authentication with Polarion.

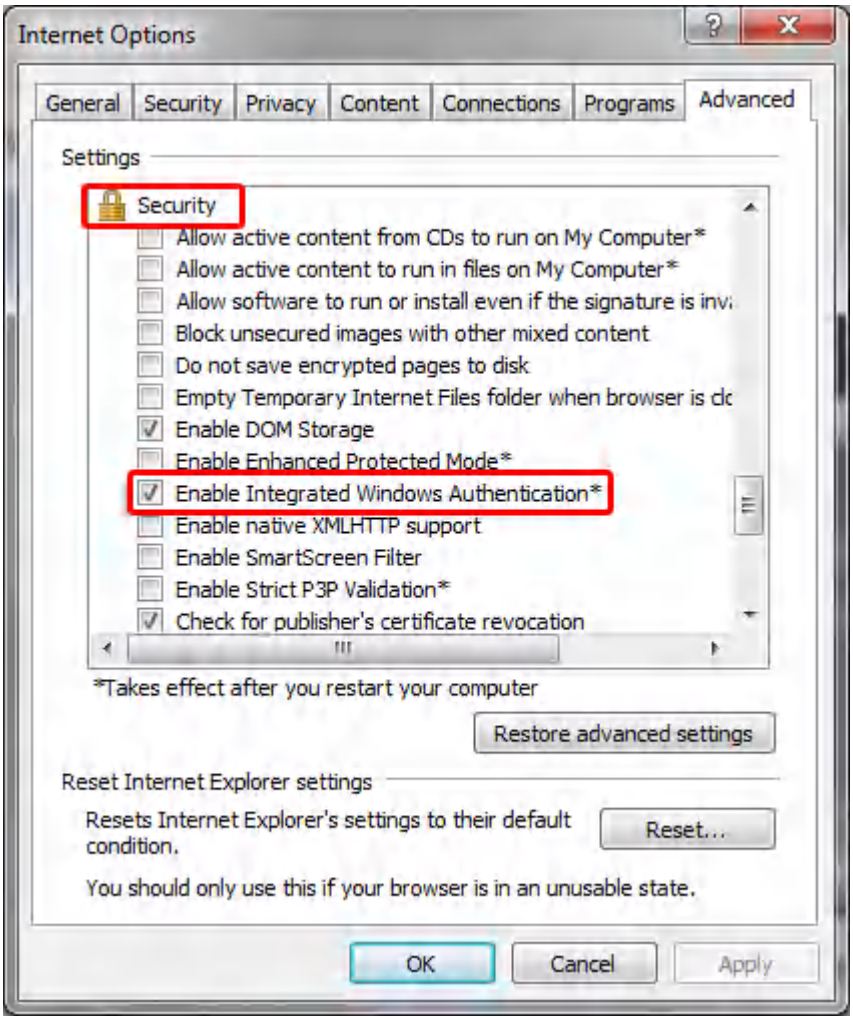
Internet Explorer 11, Chrome and Edge on Windows

All currently supported browsers (found in the Release Notes section of the README.html file), can be configured to use Kerberos authentication with Polarion.

The above browsers are configured via **Internet Options**.

(Accessible from either the Windows Control Panel, or via **Internet Explorer settings**.)

1. Enable Integrated Windows Authentication in **Internet Options** → **Advanced** → **Security**.

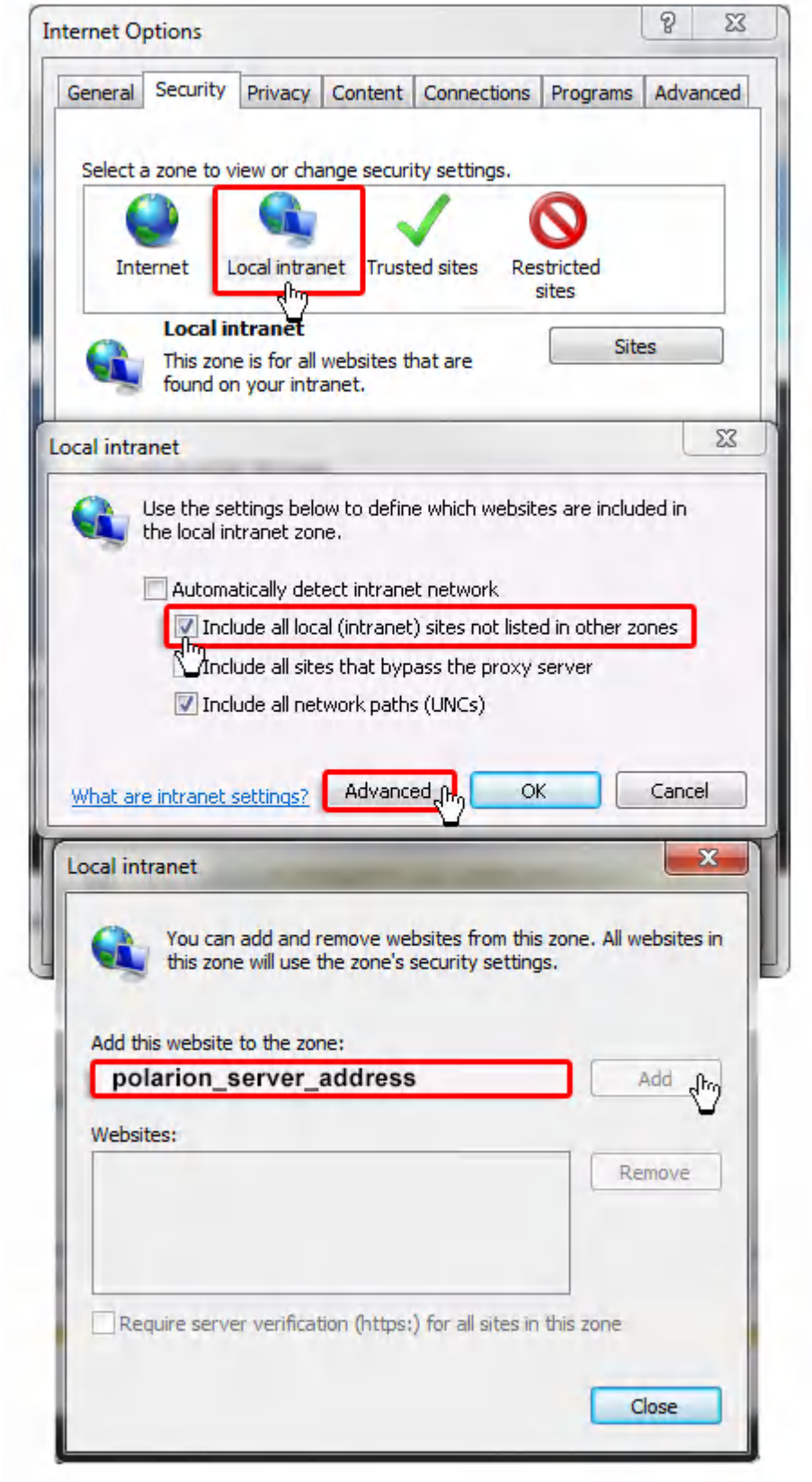


2. Ensure that your Polarion server is a member of the Local intranet sites in

Internet Options → Security→ Local intranet→ Sites.

a. Check **Include all local (intranet) sites not listed in other zones**.

b. Add it explicitly among the site on the **Advanced** tab.



Chrome on Linux

1. Create a directory **/etc/opt/chrome/policies/managed/** and within it drop a JSON file such as *yourdomain.com.json* with the following contents:
   { "AuthServerWhitelist": "*.yourdomain.com",
   "AuthNegotiateDelegateWhitelist": "*.yourdomain.com" }

2. Run Google Chrome.

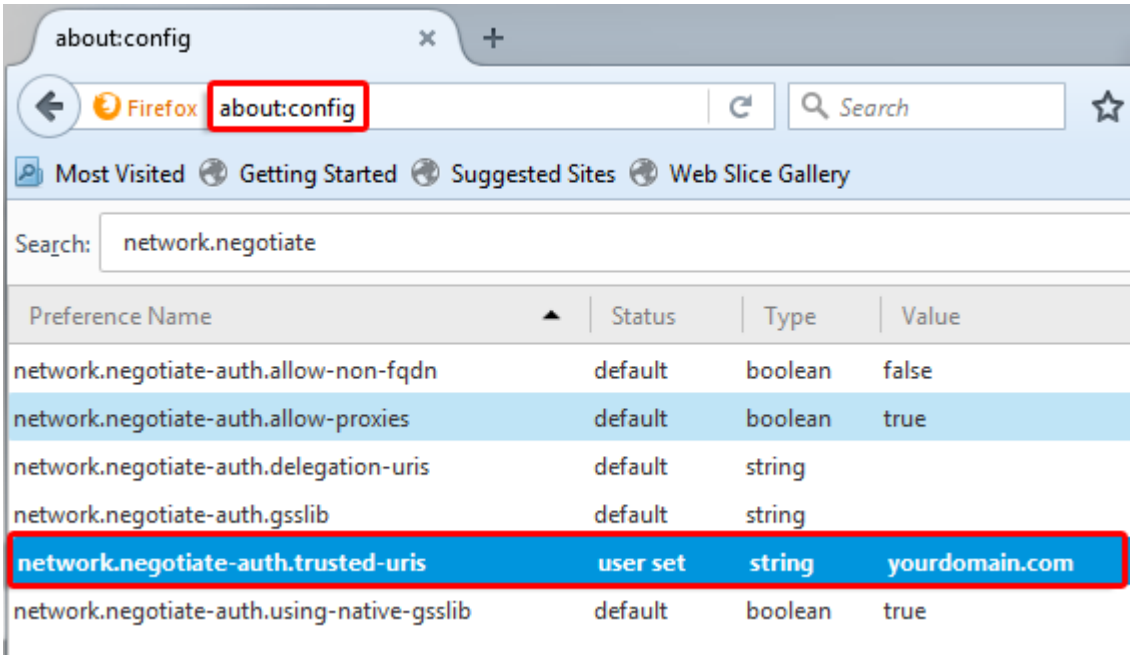
3. Open your Polarion URL *http://polarion-server.yourdomain.com/polarion* and the user is logged into Polarion automatically.

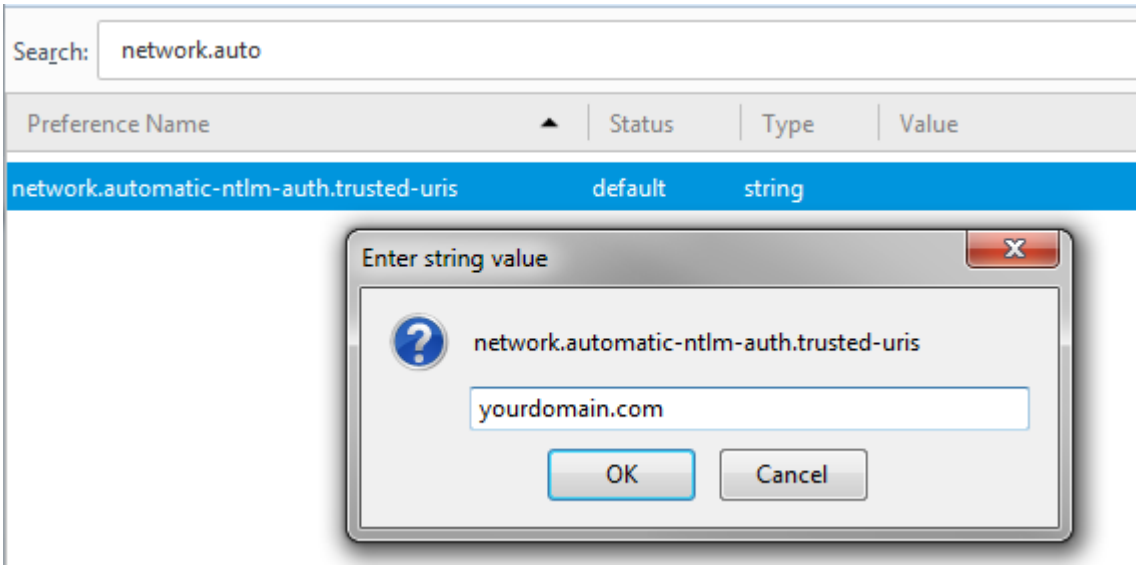
Firefox

1. Open Firefox settings by typing **about:config** in the address bar.

2. Locate the following preferences:
   network.negotiate-auth.trusted-uris
   network.automatic-ntlm-auth.trusted-uris

3. Set them by double-clicking (or right clicking and selecting **Modify**) to the name of your domain, or the Polarion server's hostname (e.g. *polarion.yourdomain.com*).





Authentication fallback for external users using Kerberos

A typical setup for enterprise customers has two user types with different authentication methods:

- **Internal Users**: Operate from within the network and are authenticated in the domain.
- **External Users**: Customers or contractors who are not authenticated in the domain, but access the internal network via a VPN connection.

Polarion supports both user types while still leveraging the benefits of SSO.

Administrators can set up an authentication fallback in a Kerberos SSO environment.

When the authentication fallback is configured, users are presented with the standard Polarion login screen when they do not provide a Kerberos authentication token.

Configure the Authentication fallback:

1. *Setup Polarion for Kerberos*.
2. Configure the connection to the same LDAP server used to authenticate users in the Kerberos via LDAP Configuration administration.
3. Set the following property to **true** in the **polarion.properties** file.
   com.siemens.polarion.security.auth.fallback=true

Tip:

When using Internet Explorer, Edge or Chrome, users may be prompted for credentials if the **Use Integrated Windows Authentication** option is enabled in **Internet Options** and Polarion is configured for Kerberos SSO. If a user cannot be authenticated in the domain to get the correct Kerberos ticket, then they can simply discard the prompt for credentials three times and will then be redirected to the Polarion login screen.

4. Teamcenter SSO

Authentication procedure flow

Polarion supports Teamcenter Security Services (TcSS) authentication.

The authentication procedure flow is the following:

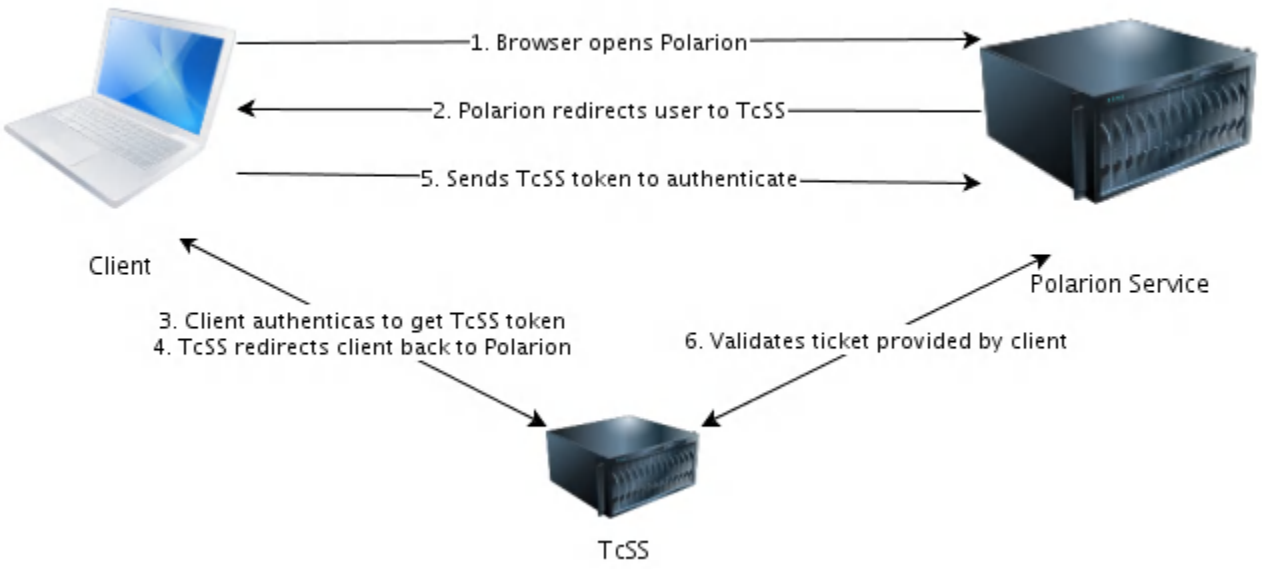
1. The client machine connects to Polarion and if a user is not authenticated, Polarion redirects them to the TcSS login service.

2. A user needs to be authenticated in TcSS to obtain an SSO token and then the TcSS login service redirects them back to Polarion.

3. The browser provides this SSO token when negotiating the authentication with Polarion.

4. The Polarion server validates the token with the TcSS identity service.

5. If the identity service successfully validates the token, Polarion authenticates the user and logs them in.



Prerequisites - TcSS configuration

A set of prerequisites is required in order to configure Polarion to authenticate using TcSS tokens:

1. Teamcenter Security Services needs to be installed on the network and connected to the company's directory service. See the **Security Services Installation/Customization documentation**.

2. The following services need to be installed and accessible by the Polarion server and client machines:
-**SSO Login Service** on e.g. **http://tcss.yourdomain.com/ssoLogin**
-**SSO Service on** e.g. **http://tcss.yourdomain.com/ssoService**

3. Configure an application ID, for example PolarionProd, for Polarion in the Identity Service registry, that points to the Polarion server's hostname.
(*http://polarion.yourdomain.com/*)

   > Note:
   >
   > Do not append **/polarion** - just point to the root.

   > Note:
   >
   > The URL configured in the Identity Service registry must match what's configured in Polarion's **base.url** in the `polarion.properties` file, or you'll have issues with the Teamcenter Connector.

4. Check that these services work correctly as described in the installation documentation above and that a user is able to login at **http://tcss.yourdomain.com/loginservice**.

Polarion configuration

Configuring Polarion to use TcSS authentication only requires that the following system properties are set in the system configuration file:

/opt/polarion/etc/polarion.properties

com.siemens.polarion.security.auth.method=tcss

com.siemens.polarion.security.tcsso.serviceUrl=http://tcss.yourdomain.com/ssoService

com.siemens.polarion.security.tcsso.loginUrl=http://tcss.yourdomain.com/ssoLogin

com.siemens.polarion.security.appId=Polarion (The default value is "Polarion".)

Once you have successfully configured the TcSS services and the Polarion server, users will not see Polarion's login page, but will be immediately redirected to the TcSS login page where they can enter their credentials. After authenticating in TcSS, an SSO session will be created and users will be redirected back to Polarion and logged in as the same user with the same login ID.

5. Other SSO considerations

Subversion access

All primary Polarion data is stored in the Subversion repository. In order to ensure the traceability of commits to their authors, the ability to write to the Subversion still needs to be subject to authentication, even when SSO authentication is used with Polarion.

Because subversion does not support standard SSO mechanisms, Polarion manages the credentials in the subversion **passwd** file for SSO authenticated users.

The following requirements on subversion access need to be met for SSO authentication setup:

1. Polarion is configured to access the Subversion repository on behalf of users via the http protocol by default. This configuration is controlled by the following system property: **repo=http://polarion-server.yourdomain.com/repo**

2. Apache needs to use file authentication, like the passwd file, for the repository location as the primary authentication source. This is configured in the**polarionSVN.conf** file of the Apache configuration.
   ```
   <Location /repo>
   ...
   AuthBasicProvider file
   </Location /repo>
   ```

3. The Polarion configuration needs to point to Apache's passwd file via the following system property: **svn.passwd.file=$[com.polarion.data]/svn/passwd**

> Note:
>
> A Polarion update sometimes requires direct access to the Subversion repository in order to adjust or add new configuration files. For this purpose, you will need at least one user with direct access to the repository using known credentials in the password file.

Direct access to subversion

For direct access to the subversion repository via the existing **/repo** location, configure this location to only authenticate via LDAP by removing the file authentication directive from:

```
[Apache configuration]/conf/extra/polarionSVN.conf

<Location /repo>

...

AuthBasicProvider file ldap

...

</Location /repo>
```

Then configure a second access location for internal Polarion access that will use the authentication via the password file that is being managed by Polarion.

```
<Location /repo-internal>

# Enable Web DAV HTTP access methods

DAV svn

# Repository location

#SVNPath "C:/Polarion/data/svn/repo"

# Write requests from WebDAV clients result in automatic commits

SVNAutoversioning on

# Our access control policy

#AuthzSVNAccessFile "C:/Polarion/data/svn/access"

# No anonymous access, always require authenticated users

Require valid-user

# How to authenticate a user. (NOTE: Polarion does not currently support
HTTP Digest # access authentication.)
```

```
AuthType Basic

AuthName "Subversion repository"

#AuthUserFile "C:/Polarion/data/svn/passwd"

AuthBasicProvider file

</Location>
```

Features requiring special configuration in an SSO setup

In a typical SSO setup, access to the server needs to be authenticated by a secure token. If there are also clients that are not able to obtain this token, Polarion provides a fallback authentication via credentials that need to be explicitly allowed for the following features. As a prerequisite, Polarion needs to be connected to the LDAP server to validate the user credentials.

1. Access to webservices from external client in SSO environment.

2. e-signatures in Kerberos SSO environment.

Subversion repository access during Polarion update

In some cases the Polarion update script does changes to the subversion repository if there are new required configuration files, or a modification of the existing configuration is required. The administrator executing the update needs to provide the credentials of a user that is capable of committing these changes to the subversion repository.

At least one service account registered in the subversion's password file with a known password and with write access to all projects in the repository is required. A new user account can be added to the subversion password file using the **htpasswd** tool provided with Apache.

Use this account to commit the changes to the subversion repository during the update.

Internal log-on pop-up behavior

In case of SAML SSO and TeamCenter SSO the logging process is redirected to the IdP login or the Team Center Login. Since it may be a security issue to embed the login page in the internal login popup IFrame, by default, TC and SAML SSO have internal login in IFrame disabled. This means, that the popup will prompt you to log in in a new tab, which will then close and you can continue working in the original tab.

However, if for any reason you want to enable the login form in the internal login popup, you can set this property in the **polarion.properties** file:

com.siemens.polarion.security.disableIFrameLogin=false

> Note:
>
> Before setting this property, make sure you IdP support the login form in an IFrame. For example, ADFS does not support this, and so it will be not possible to log in via internal popup if the IFrame is disabled.

6. Troubleshooting

SSL

To connect to the Polarion server using an HTTPS connection, make sure that Java trusts the certificate used to encrypt the communication. This especially applies to the Teamcenter Security Services Setup, Where the TcSS web application is deployed to Tomcat that runs on Java.

(See the **Importing a Certificate to the Java Keystore** section in Polarion's Help for details.)

Enable logging of SAML response

It may prove useful during the configuration of SAML SSO to be able to see the response send to polarion from IdP. To enable logging of this response, add this line to log4j.properties file located in *<Polarion_installation_folder>\polarion\plugins\org.apache.log4j_1.2.17.2015-05-04*

log4j.logger.com.polarion.platform.internal.security.auth.saml.SamlLoginClient=DEBUG,A1

After restarting Polarion, the SAML response will be visible in the main logs.

Enable international file encoding support

Support for national characters in usernames may require additional configuration. By default, Polarion encodes the Apache passwd and access files using the default Java file encoding, that matches the default OS system encoding. This default encoding may not be compatible with the character set used in LDAP. To discover what this default is set to, check for the **file.encoding** property, for example *file.encoding=Cp1252* for Windows, in the Polarion logs.

To start using national characters in usernames with SSO, first configure Polarion to use right charset.

- On Windows: Cp1252 or similar for non-Latin languages is recommended.
- On Linux: This is typically UTF-8.

1. Stop Polarion.
2. Configure the following Polarion properties:
   ```
   com.polarion.platform.repository.passwdFileEncoding=CHARSET
   com.polarion.platform.repository.accessFileEncoding=CHARSET
   svnkit.http.encoding=CHARSET
   ```
3. Manually convert the existing access and passwd files to the target character set if the original and target character sets are not compatible. Without these steps, your existing files will be corrupted the next time they are modified by Polarion.
4. Start Polarion.

Kerberos hints

While implementing SSO in Polarion, we ran into several pitfalls. In the majority of cases they were caused by a misconfigured Kerberos environment. (Be it principals, DNS, keytab, or the encryption type.) We provide details on a failed login page to aid in possible troubleshooting.

As Polarion leverages standard Java JGSS implementation, **Oracle's JGSS Troubleshooting Tutorial** is a good source for general troubleshooting information.

Client-side troubleshooting

In some cases the browser may be sending an NTLM token instead of a Kerberos token. Polarion will report this failure with the following exception:

Message Example:

Unexpected failure during authentication

javax.security.sasl.SaslException: Unsupported authentication mechanism NTLM

There are a few common cases that cause this behavior:

1. **The Client has an old Kerberos token**.
   If you configure Polarion Kerberos authentication, it is essential that the the client machine sends the correct Kerberos token. To fix this problem, make sure to logout the client from the domain and login again to get a new Kerberos token after the reconfiguration.

2. **Host name of the Polarion server does not match the principal name.**
   We recommend that the Polarion service principal uses the FQDN of the machine as its name. Make sure to access the Polarion server via FQDN to make the client send the Kerberos token. Also check that the Polarion service principal is using the HTTP prefix.

3. **Polarion server is not considered a member of the domain.**
   Integrated Windows Authentication is used only within the domain. If your Polarion server is not considered to be a member of the same domain by the client, the client will not send the Kerberos token. Check if your browser is configured correctly for Kerberos authentication. Add the Polarion server to the **Local Intranet** zone as outlined in the Client Configuration section of this guide.

4. **Wrong encryption set for the client principal.**
   The principals need to use AES 128 bit encryption. If another encryption type is set for the principals, then the client may not obtain the Kerberos token and will try to send the NTLM token instead. Make sure to configure your principals to use AES 128 bit encryption.

Server-side troubleshooting

Typical problems in a Kerberos setup are cases when the keytab with a password for the Polarion service principal is not exported properly or not understood by Java. Unfortunately, the Java Kerberos implementation does not report these issues in a very clear way. A problem with the keytab results in Polarion being unable to login into the KDC in order to validate the token obtained from the client. In such cases, a variety of messages may appear including:

Message Example:

Unexpected failure during authentication

javax.security.sasl.SaslException: Could not initiate login

There are a few common cases that cause this behavior, and the majority of them can be troubleshot using **Oracle's JGSS Troubleshooting Tutorial**. The most typical cases are listed below:

1. **Keytab was not exported correctly**.
   Check that the principal, their password, the mapped user, and the encryption type are set correctly when exporting the keytab. See the example below and refer to the correct values for each ktpass command attribute.
   Use the Kinit tool provided with the standard Java installation to check whether the Kerberos login and keytab are correct.
   Example:
   kinit -k -t C:\Polarion\polarion\configuration\polarion.keytab <HTTP/principal>

2. **Incorrect configuration of the Polarion service principal.**
   Check that the encryption type configured for the Polarion service principal is AES 128 bit.
   In Windows environments, the event log on the KDC may provide more details about potential problems when exporting the keytab, because they are not logged by the ktpass tool.

3. **Java cannot load the exported keytab file.**
   In specific cases, the keytab may not be understood by Java when exported directly from the KDC. If that is the case, use the ktab tool that is part of the standard Java installation.