NevisFIDODeployable_client

Enter the ID of the nevisIDM `Client`.

Only 1 client is supported.

NevisAuthRealmBase_cookieName

Each realm has its own session cookie. By default, this cookie will be called `Session_<pattern-name>`

Set this optional property to use a different name (e.g. `ProxySession`).

Note that each realm has its own session. However, if the same cookie name is configured for multiple realms running on the same host the sessions will be cleaned up together when the first session expires.

NevisIDMDeployable_port

Port the nevisIDM instance is listening on.

OAuth2AuthorizationServer_scopes

Enter scopes which may be requested by an `OAuth 2.0 / OpenID Connect Client`.

The scope `openid` must be allowed when `OpenID Connect` is used.

NevisIDMUserLookup_attributes

Enter user attributes to fetch from nevisIDM.

Important attributes are:

- `extId` - unique ID of the user in nevisIDM
- `loginId` - name which could be used to login (instead of email)
- `firstName`
- `name` - surname
- `email`
- `mobile`
- `language` - language stored for user (can differ from `Accept-Language` sent by the browser)

For a complete list check the documentation of IdmGetPropertiesState.

Some attributes (e.g. `extId`, `email`, and `mobile`) are always fetched as they are required by standard authentication steps.

The attributes will be stored in the user session as `ch.nevis.idm.User.<attribute>`.

Attributes may be used in sub-sequent authentication steps or included in application access tokens (e.g. `NEVIS SecToken`, `SAML Token`, or `JWT Token`).

For instance, use them in a `Generic Authentication Step` via the expression `${sess:ch.nevis.idm.User.<attribute>}`.

NevisIDMDeployable_encryptionAlgorithm

Encryption algorithm.

NevisAdaptAuthenticationConnectorStep_highThreshold

Will be considered only if `Profile` is set to either `balanced`, `strict` or `custom`.

Set the risk score threshold [0...1] for high threat.

NevisAuthRadiusFacade_responses

Configure additional Radius responses depending on your authentication flow.

For instance, configure `Access-Challenge` responses if your authentication flow is interactive.

Response rules configured here are evaluated first. You can therefore overrule the default rules added by this pattern.

No configuration may be required for basic username / password login as username and password can be sent by the Radius client in the initial `Access-Request` message.

UserInput_onSuccess

Configure the step to execute after the user has provided input. If no step is configured here the process ends with `AUTH_DONE`.

SamlSpConnector_context

Select `nevis` if the SAML service provider is provided by a `SAML SP Realm` and you want to use `Authorization Policy` to specify the required `Authentication Level` for application protected by that realm.

When `nevis` is selected the roles and attained authentication level are added to the SAML `Response` via an `AuthnContextClassRef` element.

Example:

```
<saml2:AuthnStatement AuthnInstant="2021-05-07T06:48:14.967Z">
  <saml2:AuthnContext>
    <saml2:AuthnContextClassRef>...,nevisIdm.Admin,urn:nevis:level:1</saml2:AuthnContextClassRef>
  </saml2:AuthnContext>
</saml2:AuthnStatement>
```

Select `PasswordProtectedTransport` to add the following standard context:

```
<saml2:AuthnStatement AuthnInstant="2021-05-07T06:48:14.967Z">
  <saml2:AuthnContext>
    <saml2:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</saml2:AuthnContextClassRef>
  </saml2:AuthnContext>
</saml2:AuthnStatement>
```

Select `none` to not add any `AuthnContext` element.

NevisMetaWebApplicationAccess_backendTrustStore

Assign the Trust Store provider for outbound TLS connections. If no pattern is assigned a trust store will be provided by nevisAdmin 4 automatic key management.

NevisProxyObservabilitySettings_traceExporterAddress

Enter the target URL (`host:port`) of the backend services to which the exporter is going to send spans. The `/v1/traces` path is automatically attached to it.

CustomNevisMetaLogFile_levels

Configure log levels.

See the nevisMeta Technical Documentation, chapter Logging for details.

Hint: If you only change log levels nevisAdmin 4 does not restart the component in classic VM deployment. The new log configuration will be reloaded within 60 seconds after deployment.

Examples:

```
ch.nevis.nevismeta = INFO
ch.nevis.ninja = DEBUG
```

NevisAdaptDeployable_addons

Assign an add-on pattern to customize the configuration.

HeaderCustomization_requestHeaders

Adds/overwrites HTTP headers in requests.

The syntax is: `<header name>:<value>`

Examples:

```
X-Forwarded-For: ${client.ip}
User-ID: ${auth.user.auth.UserId}
```

Note: change the `Filter Phase` to replace headers early / late.

In order to use the `${exec: ...}` syntax of nevisProxy for passwords, use an inventory secret to skip the validation of the value.

OAuth2Scope_consentRequired

Select `enabled` if consent shall be requested for this scope.

ActiveMQClientConfiguration_messageBrokerURL

Set the URL for the ActiveMQ message broker. Example:

```
ssl://my-message-broker:61616
```

LogSettingsBase_maxBackupIndex

Maximum number of backup files to keep in addition to the current log file.

DummyLogin_buttons

Assign an `Dispatcher Button` to add a button which points to a different authentication step.

NevisAdaptPluginPattern_propagateGeolocation

Risk scores to be delivered to the client in the request headers. This option configures enables geolocation risk score to be propagated.

HostContext_tls

Choose between:

- `recommended` : for high security, apply the recommended settings for `SSLProtocol` and `SSLCipherSuite` . The settings may change in future releases. Check the nevisProxy Technical Documentation for details. This works with modern browsers and clients. Current `recommended` values are:

```
sslProtocol = '-all +TLSv1.2 -TLSv1.3'
sslCipherSuite = 'ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-RSA-AES256-GCM
```

- `compatible` : the `SSLProtocol` and `SSLCipherSuite` will be based on Mozilla's SSL configuration for Apache server. These settings provide high compatibility with older browsers and clients. Current `compatible` values are:

```
sslProtocol = '-all +TLSv1.1 +TLSv1.2 -TLSv1.3'
sslCipherSuite = 'ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256
```

- `custom` : assign a `TLS Settings` pattern via `Additional Settings` to apply your own configuration. If not provided, `SSLProtocol` and `SSLCipherSuite` follow the `recommended` settings.

PemTrustStoreProvider_rootDirName

Set to deploy the trust store underneath a *base* directory. The trust store will be established at:

```
/var/opt/keys/trust/<base>/<name>
```

This configuration may be used to prevent trust stores overwriting each other and is only required in complex setups with multiple projects or inventories.

NevisAdaptEvent_minMatch

Specify the minimum number of matching risk events to continue with `Authentication Step`. Picking a number that exceeds the size of selected `Risk Events` will set `all` during generation.

NevisIDMClientCertAuthentication_onSuccess

Assign an optional step to execute after successful authentication.

NevisIDMClient_displayNames

The name of the client in different languages.

The format is:

- two letter language code in lower case
- separator characher: `=` or `:`
- the client name in that language

For example:

```
de:Beispiel-Client
fr:Exemple de client
```

CustomNevisIDMLogFile_regexFilter

If set, messages for `application.log` which match the given regular expression won't be logged.

The regular expression must match the entire line. For instance, you may use the following format to match `some text`:

```
.*some text.*
```

NevisAdaptUserNotification_onSuccess

Set the step to continue with on successful authentication.

ServiceBase_realm

Optionally assign a realm to protect this application or service.

NevisIDMDeployable_encryptionKey

Enter an encryption key as Base64. This is mandatory because of security.

For existing setups please enter the value of `security.properties.key` from your `/var/opt/nevisidm/<instance>/conf/nevisidm-prod.properties` file.

If you don't know which value was used so far you may generate a new key and set `Encryption Fallback` to `enabled` to ease migration.

When there are no URL tickets or encrypted properties the fallback can be disabled.

For new setups the key should consist of random bytes. The following openssl command generates a random key and returns the Base64 value:

```
openssl rand -base64 16
```

Note that when `Encryption Algorithm` is set to `AES`, the key length must be 8, 16 or 24 bytes. 8 byte long `AES` keys are strongly discouraged for new instances, but supported for legacy instances.

HeaderCustomization_responsePhase

- `BEFORE_SANITIATION` - manipulate request headers late to also cover any headers set by nevisProxy.
- `AFTER_AUTHENTICATION` - default behaviour which should work in most cases.
- `END` - manipulate response headers early hiding them from other nevisProxy filters which operate on responses.

HeaderCustomization_condition

Set to do the header customization only if the given condition applies.

The condition is checked for `Add / Overwrite Headers` on requests and on responses.

You can use the expressions mentioned above.

Syntax:

```
${expression} == value
```

Examples:

```
${request.header.Content-Type} == application/x-www-form-urlencoded
```

NevisAuthDeployable_classPath

Enter directory paths to be added to the `classPath` of the `AuthEngine` .

The paths will be added as a prefix. Entries added by other patterns (e.g. `/opt/nevisidmcl/nevisauth/lib` or `/opt/nevisauth/plugin`) are preserved.

This is an advanced setting which should only be used when working with custom `AuthState` classes.

GoogleLogin_clientId

ClientID is `Client ID` provided by Google when you create a OAUTH 2.0 credential in Google.

NevisDPLogSettings_maxFileSize

Maximum allowed file size (in bytes) before rolling over.

Suffixes "KB", "MB" and "GB" are allowed. 10KB = 10240 bytes, etc.

This configuration applies to non-Kubernetes deployment only.

Note: not relevant when rotation type is `time`.

CSRFProtectionSettings_sameSite

Set to `lax` to issue a separate cookie with the `SameSite` flag set to `lax`. In this configuration, links and redirects from other domains are allowed, while CSRF-prone requests (e.g. `POST`) should be prevented by the browser.

Set to `off` to not send an additional cookie. There are several reasons why this feature may be disabled:

- Not all browsers support the `SameSite` flag and behave incorrectly by never sending the cookie. Older versions of IE and Windows may be affected.

- The `SameSite` flag breaks SAML use cases when POST binding is used. SP-initiated authentication does work with NEVIS but all other SAML process (e.g. logout) will fail.

ICAPScanning_contentType

Optional property to restrict scanning to a certain Content-Type (regular expression is supported here).

Example: `application/.*`

NevisProxyDatabase_encryption

Enables TLS in a specific mode. The following values are supported:

- `disabled` : Do not use TLS (default)
- `trust` : Only use TLS for encryption. Do not perform certificate or hostname verification. This mode is not recommended for production applications but still safer than `disabled` .
- `verify-ca` : Use TLS for encryption and perform certificates verification, but do not perform hostname verification.
- `verify-full` : Use TLS for encryption, certificate verification, and hostname verification.

LuaPattern_phase

Defines the position of the filter-mapping for this Lua filter. Which position to choose depends on your use case.

For requests filters will be invoked from `START` to `END` . For responses filters will be invoked from `END` to `START` .

Choose from the following filter phases:

- `START` : applied as early as possible for requests and as late as possible for responses.

- `BEFORE_SANITATION` : applied before filters which validate the request (e.g. Mod Security).
- `SANITATION` : used for security. This is the first phase which allows accessing the session for applications protected by a realm.
- `AFTER_SANITATION` : your request has passed security checks.
- `BEFORE_AUTHENTICATION` : applied just before authentication.
- `AUTHENTICATION` : used by the filter which connects to nevisAuth for applications which are protected by an `Authentication Realm` .
- `AFTER_AUTHENTICATION` : the request has level 1 authentication. Used by `Authorization Policy` for `Authentication Level` stepup.
- `BEFORE_AUTHORIZATION` : choose this phase to do preprocessing before authorization.
- `AUTHORIZATION` : used by `Authorization Policy` for `Required Roles` check.
- `AFTER_AUTHORIZATION` : used by patterns assigned as `Application Access Token` to applications.
- `END` : applied as late as possible for requests and as early as possible for responses.

TANBase_title

Change the Gui title.

We recommend to enter a label here and provide translations for this label in the `Authentication Realm` .

UserInformation_label

Enter a label or an expression for the text message that shall be presented to the user.

Translations for the label can be defined in the realm pattern.

If not set the expression `${notes:lasterrorinfo}` is used.

GenericDeployment_executableFiles

Expression to select files which shall have the *executable* flag. Add exact file names or `*.<ending>` .

Example:

- myScript.sh
- *.py

SOAPServiceAccess_schema

Optional property to upload a schema.

This feature is experimental and may change in future releases.

You must upload all required XSD schema files. Each XSD schema file must declare 1 target namespace which will be extracted from the first `targetNamespace` attribute found in the file.

Upload of a WSDL file is optional. If provided, the WSDL must contain a `types` declaration containing an XSD schema. However, this schema definition can be empty. Here is a minimal example:

```
<types>
   <xsd:schema targetNamespace="urn:com.example:echo"
               elementFormDefault="qualified">
   </xsd:schema>
</types>
```

The actual schemas must still be uploaded as separate files.

SamlResponseConsumer_idp

Assign a `SAML IDP Connector` for each SAML Identity Provider.

SP-initiated authentication is not supported and thus the `Selection Expression` of the connector patterns is ignored.

HostContext_resources

Upload a ZIP to provide your own resources.

By default, the following resources are provided:

- `/favicon.ico`
- `/index.html`
- `/errorpages/403.html`
- `/errorpages/404.html`
- `/errorpages/500.html`
- `/errorpages/502.html`
- `/resources/logo.png`
- `/resources/bootstrap.min.css`
- `/resources/default.css`

This host has its own error handler (`ErrorHandler_Default`) which is assigned to the root location (`/*`). The error handler will replace the response body when an HTTP error code occurs and an error page is available.

Error pages for HTML must be added the sub-directory `errorpages` and named `<code>.html` .

The error code is returned to the caller as this may be required by some REST clients.

If you do not want this you can assign a specific `HTTP Error Handling` pattern to this `Virtual Host` or to applications via `Additional Settings` .

The servlet hosting the above resources is usually mapped to the root location (`/*`), however if there is already another servlet mapped there, the servlet is mapped to individual root files and directories.

If there is an undesired mapping, it can be deleted by removing the given resource from the zip file.

NevisProxyObservabilitySettings_traceContextInjection

Choose one of:

- **enabled**: inject the current context (span ID, trace ID, etc) as a HTTP header to the request
- **disabled**: do not inject the current context in the request

NevisDetectEntrypointDeployable_contentType

Apply restriction based on request header Content-Type

NevisAdaptAuthenticationConnectorStep_fingerprintJsVersion

This configuration option gives the administrator the ability to ensure backwards compatibility in case so far V2 fingerprints have been in use.

- `V2` - to ensure backward compatibility, FingerprintJS V2 will be used
- `V3` - default option, uses FingerprintJS V3

NevisFIDODeployable_customURILink

Custom URI links will open the mobile app directly.

The scheme must be registered for the mobile app.

See Link Structure for Custom URIs for details.

If the mobile app is not installed an error will occur.

Example:

```
myaccessapp://x-callback-url/authenticate
```

NevisMetaServiceAccessBase_token

Assign a `NEVIS SecToken` pattern.

The token informs nevisMeta about the authenticated user.

If you are not using automatic key management then you also have to configure `nevisMeta Instance` / `NEVIS SecToken Trust` so that the signer certificate is trusted.

NevisAuthDeployable_backendKeyStore

Assign the Key Store provider for outbound TLS connections. If no pattern is assigned a key store will be provided by the nevisAdmin 4 PKI.

HostingService_rewrites

Rewrite rules for serving files.

This can be useful if a file should be served under a different name, or to map extensions to file names.

Examples:

Source	Destination
`/static/picture`	`/static/picture.jpg`

PemTrustStoreProvider_truststoreFile

Upload trusted certificate(s) in PEM format.

If you set a *variable*, the variable should be a list of secret file references in the inventory. Example:

```
my-variable:
  - inv-res-secret://147cc54a5629fadac761ec01#some-cert.pem
  - inv-res-secret://147cc54a5629fadac761ec01#some-other-cert.pem
```

Upload files for this variable by clicking `Attach files` in the drop-down on the inventory screen.

If you are deploying to Kubernetes you may store the trust store content in a Kubernetes secret. You can pick any name for the Kubernetes secret but the keys must be as in the following example:

```
my-variable:
  - k8s-secret-file://dummy-truststore:truststore.pem/
  - k8s-secret-file://dummy-truststore:truststore.jks/
  - k8s-secret-file://dummy-truststore:truststore.p12/
  - k8s-secret-file://dummy-truststore:keypass/
```

Note that nevisAdmin 4 does not notice when the content of the Kubernetes secret changes. Manual interaction (terminating pods) is required in that case.

NevisIDMUserLookup_unitProperties

Enter unit properties to fetch from nevisIDM and store in the unit session.

Properties must be created in the nevisIDM via SQL.

HostContext_securityHeaders

Configure security response headers:

- `off` does not set any security headers
- `basic` sets default headers on responses. That is:
  - `Strict-Transport-Security: max-age=63072000`
  - `X-Content-Type-Options: nosniff`
  - `Referrer-Policy: strict-origin-when-cross-origin`
- `custom` configure `Security Response Headers` via `Additional Settings`

OAuth2Client_pkceMode

The following types of PKCE modes are supported:

- `allowed` (default): If the client sends PKCE information in the form of a code challenge in the authorization request, the code challenge will be validated. If the code challenge is not valid, the authorization will fail. But if no code challenge is included in the authorization request, the authorization will not fail.
- `required` : The client must send valid PKCE information. If no code challenge is included in the authorization request, the authorization will fail.

- `s256-required` : The client must send valid PKCE information using the S256 code challenge method. The authorization will fail if no code challenge is included in the authorization request, or if the code challenge does not use the S256 code challenge method.

If the client supports the s256 code challenge method, then `s256-required` is the recommended value.

NevisAdaptDatabase_password

Provide the DB password here.

GenericIngressSettings_tlsSettings

If `disabled` , the TLS related settings are removed from the generated Ingress resource, which means the default certificate provided by NGINX will be used for the TLS termination.

It's only recommended to use this option, when an additional loadbalancer is used in front of NGINX (e.g. Cloudflare), which already provides a valid certificate.

CustomProxyLogFile_eventLog

Enable event logging capability of nevisProxy.

Event logs are not forwarded to syslog.

NevisIDMUserCreate_unitId

Enter the unit ID where the user shall be created.

FIDO2Onboarding_userVerification

User verification is a crucial step during WebAuthn authentication process as it confirms that the person attempting to authenticate is indeed the legitimate user.

This setting allows to configure the user verification requirements for onboarding.

Allowed values:

- `discouraged`
- `preferred`
- `required`

NevisFIDOLogSettings_serverLogFormat

Log4j 2 log format for the default SERVER logs. This pattern is used for **non**-kubernetes deployments.

Note: not relevant when Log Targets is set to `syslog`.

NevisAdaptDeployable_observationConfig

Used to assign a `nevisAdapt Observation Cleanup Configuration` pattern to configure the time interval for cleaning up observation data.

This is an optional setting. Default values if nothing is set:

- Observation Timeframe: `60d`
- Trusted Cleanup Period: `1d`
- Untrusted Cleanup Timeframe: `12d`

NevisAuthDeployable_startTimeout

Enter a timeout to wait for nevisAuth startup.

Set a higher value if nevisAuth takes longer to start.

This setting applies to classic VM-based deployment only.

NevisProxyObservabilitySettings_metricsMode

Choose one of:

- **enabled**: enable the metrics feature of OpenTelemetry
- **disabled**: disable the metrics feature of OpenTelemetry

NevisIDMUserLookup_properties

Enter user properties to fetch from nevisIDM and store in the user session.

Properties must be created in the nevisIDM via SQL.

SAPLogonTicket_authScheme

Authentication scheme associated with this ticket. See SAP documentation of SAP SSO logon tickets for more information. Default value is SAP's default and should be correct for most cases.

ProxyPluginPattern_riskScores

Risk scores to be delivered. Please add entries in the following format:

RiskScoreName=#ColorCode

Button_buttonName

Enter a `name` to use for the `GuiElem`.

Configuration is optional but may be required when the button is rendered differently based on the name.

If missing, the sanitized name of the pattern will be used.

NevisDetectDatabase_flywayLicenceKey

Please provide a licence key in case you would use the Flyway Teams Edition.

This is recommended only in case you would use an old database version (more than 5 years old). If you do not provide a licence key, the Flyway Community Edition will be used by default.

For more information about Flyway editions please visit this page Flyway.

CustomProxyLogFile_logLevelParameters

Configure log levels.

Overrules the `Log Level` property.

See nevisProxy Reference Guide `Operation and Administration / Debugging` for possible trace groups.

Enter the **suffix** of the name of the trace group and a log level.

Supported log levels are:

- ERROR
- NOTICE
- INFO
- DEBUG
- DEBUG_HIGH
- TRACE

Do **not** enter numbers for the log level as nevisAdmin4 will calculate them automatically.

The default configuration is:

```
NavajoOp = INFO
NProxyOp = INFO
```

Debug startup:

```
NavajoStart = INFO
```

Debug `HTTP Header Customization`:

```
IW4HdrDlgFlt = DEBUG
```

AuthStatePatch_realm

Authentication realm to patch.

SamlSpRealm_postProcess

Assign a *Generic Authentication Step* to apply custom post-processing logic to an SP-initiated SAML process (e.g. authentication, session upgrade, or logout).

By assigning a step here the last AuthState of the process will be replaced so that it points to the first AuthState provided by the assigned step. This AuthState should be marked with the name `${state.entry}`.

Use the expression `${state.done}` to complete with the SAML process.

SharedStorageSettings_claimName

The name of the PersistentVolumeClaim.

For more information regarding persistent volumes in Kubernetes please visit this page

SamlResponseConsumer_host

Assign a `Virtual Host` which shall serve as entry point.

InBandMobileAuthenticationRealm_nevisfido

Assign a nevisFIDO instance. This instance will be responsible for providing the in-band authentication services.

AuthServiceBase_realm

Assign an `Authentication Realm`.

SamlIdpConnector_url

Enter the `Location` of the SAML `SingleSignOnService`. This may be a URL or a path on the same virtual host.

nevisAuth will send an `AuthnRequest` to this location to delegate the authentication or session upgrade process to the IDP.

By default, the `AuthnRequest` contains a `RequestedAuthnContext` which specifies the required authentication level. You can disable this feature via `Custom Properties`.

NevisProxyDatabase_params

Add custom `init-param` for the MySQL session store servlet.

Check the nevisProxy technical documentation for supported parameters of the servlet class `ch::nevis::nevisproxy::servlet::cache::mysql::MySQLSessionStoreServlet`.

NevisAdaptPluginPattern_propagateDeviceFingerprint

Risk scores to be delivered to the client in the request headers. This option configures enables device fingerprint risk score to be propagated.

MicrosoftLogin_claimsRequest

The claims request parameter. This value is expected to be formatted in JSON and does not accept trailing spaces nor tabs.

GenericDeployment_commandTrigger

Defines when or how often the command is executed.

Possible values are:

- `always` : Execution during each deployment.
- `onFileChange` : Executed if an uploaded file under the specified `Path` has changed.
- `onFileTriggers` : Executed if a file that is listed under `Command: Execution File Triggers` has changed.
- `onFileChange + onFileTriggers` : Combining both options above.

NevisIDMAuthorizationsAddon_roleAssignmentFile

Add properties for `rolesAssignment.properties` . If a role not defined in the uploaded file default values will be used for it.

See Data room authorization for details.

You can input the role with or without `nevisIdm` prefix. For instance, both `Root` are `nevisIdm.Root` are supported.

GenericAuthService_pathAddons

Assign add-on patterns to customize the `Frontend Path` .

NevisAuthRealmBase_logrendKeyStore

Assign a pattern which sets up a key store to use for 2-way HTTPs connections to nevisLogrend.

If no pattern is assigned no key store will be setup and 1-way HTTPs or plain HTTP will be used depending on the connection URL of nevisLogrend.

GenericDeployment_command

Bash shell expression which will be executed from the working directory `Path` as the deployment user (`__connection_user` variable in the inventory).

Example:

- ./my_script.sh

The command will run depending on the `Command: Execution` setting: always or conditional (e.g. `onFileChange`). Note that with the onFileChange setting, the command is *not* automatically executed if you change it here.

Tip: Instead of specifying your shell instruction(s) here, add them as a separate script file into `Files` . For example, if the file name is `my_script.sh` , enter `./my_script.sh` as the `Command` . This way, the script *will* be re-executed each time you upload an updated script file and deploy the project (if `onFileChange` command execution is configured below).

NevisLogrendDeployable_trustStore

Used when mutual (2-way) HTTPs is configured. If no pattern is assigned here automatic key management will provide the trust store.

RoleCheck_notFound

Assign a step to continue with when the user has **none** of the configured roles.

If no step is assigned, error code `403` will be returned in this case.

ApplicationProtectionDeployableBase_keyStore

Used when simple or mutual (2-way) HTTPs is configured. If no pattern is assigned here automatic key management will provide the key store.

ApplicationProtectionDeployableBase_clientAuth

Setting for 2-way TLS on the nevisAdapt HTTPs endpoint. There are 3 options will affect the callers (e.g. nevisProxy or technical clients accessing nevisAdapt REST APIs)

- required: Callers **must** present a client certificate.
- requested: Callers **can** present a client certificate.
- disabled: Callers **must not** use a client certificate.

The `Frontend Trust Store` must contain the issuing CA.

NevisAdaptUserNotification_notificationType

This mandatory property selects the actual communication event and thus the used template text type.

NevisAdaptDeployable_ipPrivateNetworkCountryCode

When selected 'disabled' on IP Private Network Filter, the country code of the IP address is not in the list of private network country codes.

You can also assign a default Geolocation by country code by ISO 3166 alpha-2.

GenericAuthWebService_configFile

The file should contain `WebService` elements only.

Uploading a complete `esauth4.xml` is not supported.

NevisMetaWebApplicationAccess_meta

Reference the nevisMeta Instance.

EmailInputField_optional

Input into the field is optional or mandatory.

Choose between:

- `optional` - No input is required to the field.
- `mandatory` - Input is required to the field.

NevisAdaptObservationCleanupConfig_timeframeDays

This value defines the observation lookup period, the cleanup of trusted observations cannot happen sooner than this.

The default value is `60d` .

NevisAuthRealm_onDemandFallback

Assign an authentication step which should be invoked when a session upgrade is triggered and none of the `Session Upgrade Flows` can be applied.

NevisIDMUserLookup_unitAttributes

Enter unit attributes to fetch from nevisIDM.

Possible attributes are:

- `extId` - unique ID of the unit in nevisIDM
- `state` - state of the unit in nevisIDM
- `name`
- `displayName`
- `displayAbbreviation`
- `location`
- `description`
- `hname`
- `localizedHname`
- `ctlCreDat`
- `ctlCreUid`
- `ctlModDat`
- `ctlModUid`

For a complete list check the documentation of IdmGetPropertiesState.

The attributes will be stored in the user session as `ch.nevis.idm.Unit.<attribute>` .

Attributes may be used in sub-sequent authentication steps or included in application access tokens.

For instance, use them in a `Generic Authentication Step` via the expression `${sess:ch.nevis.idm.Unit.<attribute>}` .

NevisAuthDeployable_scripts

Upload shared Groovy scripts used in your authentication steps.

You can use the expression `${var.<name>}` inside your scripts to refer to an inventory variable.

How the value of a variable is generated into the script depends on the variable content:

Scalar variables will be generated as-is. This means the variable expression has to be within a Java String (') or a Groovy *GString* (").

YAML variables of type **sequence** will be generated as a **Groovy list**.

For instance, let's assume you have the following variable in your inventory:

```yaml
my-sequence-var:
  - some-entry
  - another-entry
```

In your script you may use the expression as follows:

```
def myList = ${var.my-sequence-var}
```

The following will be generated:

```
def myList = ["some-entry", "another-entry"]
```

Inventory variables containing a **YAML mapping** are not supported yet.

Uploaded files will be deployed to the `conf/groovy` folder of the nevisAuth instance.

GenericModQosConfiguration_serverDirectives

Server level directives can be entered here.

These directives apply to the entire `nevisProxy Instance` which means that other `Virtual Host` patterns may be affected.

Examples:

```
QS_ClientEventBlockCount 200 300
QS_SetEnvIf NAVAJO_HTTPSESS_CREATED !QSNOT QS_Block=yes
QS_SrvMaxConnClose 85%
QS_SrvMaxConnPerIP 75
QS_SrvMinDataRate 75 300 250
```

TransactionConfirmation_nevisfido

Assign a `nevisFIDO UAF Instance` . This instance will provide the transaction confirmation services.

NevisIDMUserCreate_loginId

Define how the `loginId` is set:

- `auto` : the `loginId` is generated. `loginIdGenerator.enabled=true` must be set in the client policy. This can be achieved via the `nevisIDM Administration GUI` .
- `email` : use the email for the `loginId` . The `email` must be provided via `Mandatory User Attributes` .
- `value` : the `loginId` must be provided via `Mandatory User Attributes` .

PropertiesTestPattern_keyValueProperty

Enter key=value pair(s). This property also supports other separators.

NevisIDMDeployable_frontendTrustStore

Assign the Trust Store provider for the HTTPs endpoint. If no pattern is assigned the Trust Store will be provided by the nevisAdmin 4 PKI.

Webhook_url

Enter the URL to call.

GenericAuthenticationStep_nextSteps

Assign follow-up steps.

The order of steps is relevant. The first step in this list has index `1`.

You may reference a step in the configuration via the expression `${state.exit.<index>}`.

SAPLogonTicket_recipientClient

See SAP documentation of SAP SSO logon tickets for more information. Setting no value for this property should be correct for most cases.

NevisIDMChangePassword_newPassword2

Mandatory input value to use for confirming the new password if `Show GUI` is disabled and `Show Confirmation Field` is enabled.

NevisFIDODatabase_type

Choose between `MariaDB` and `PostgresSQL`.

We recommend to use `MariaDB` as it is supported by all Nevis components that have a database.

Note: `PostgresSQL` database is only experimental configuration.

GenericIngressSettings_nodePortService

If `enabled`, the generated services for the Ingresses will be of type NodePort. This allows direct connection to the nevisProxy instance.

NevisIDMPasswordLogin_buttons

Assign an `Dispatcher Button` to add a button which points to a different authentication step.

NevisDetectDatabase_database

Enter the name of the database.

This database will be created in the database service.

TANBase_buttons

Assign a `Dispatcher Button` to add a button which points to a different authentication step.

NevisAdaptAuthenticationConnectorStep_clientKeyStore

The key store used by this pattern to establish a connection with the nevisAdapt component. For a client TLS connection, this key store should be trusted by the `nevisAdapt Instance`. If no pattern is assigned here automatic key management will provide the key store.

OAuth2Client_address

Set to `allowed` to allow this client to request the scope `address`.

This scope produces various claims.

AuditChannel_channelClass

Enter the fully qualified name of your Java class. The class must implement the AuditChannel interface.

The class must be on the classpath of the `AuthEngine`. You can upload your JAR file in the `Classloading` tab of the `nevisAuth Instance` under `Custom Dependencies`.

GenericDeployment_files

Upload the files which will be copied into the `Path`.

To upload files into subdirectories within `Path`, add a single .zip file with files and directories. Unpacked files will have `Owner` and `Group` applied. Note: If multiple files are uploaded, any .zip file is deployed as is, without being extracted.

It is not supported to overwrite files generated by other patterns. See also `Path` above.

NevisAuthRadiusFacade_realm

Assign a nevisAuth Realm which shall be exposed via Radius.

NevisFIDODeployable_noUserVerificationTimeout

Maximum time that a FIDO2 client has to send the response in a ceremony where user-verification is **not** required.

Default value is 2 minutes.

RequestValidationSettings_scope

Sets the scope of request validation:

- `all` : validation will be applied to all requests. This includes authentication.
- `backend` : validation will be applied to requests which are sent to the backend application. The authentication is excluded.
- `authentication` : validation will be applied to requests which are sent to nevisAuth.

TCPSettings_keepAliveInactiveInterval

Inactivity duration allowed before a TCP connection is dropped. By leaving this field empty, you will be using the nevisProxy default value.

NevisAdaptDatabase_parameters

Enter parameters for the DB connection string.

Enter 1 parameter per line.

Lines will be joined with `&`.

The default is:

```
useMysqlMetadata=true
```

The default value will be used **only** when no parameters are entered.

If you want to keep the default parameters, add them as well.

NevisAdaptDatabase_hikariValues

Specify custom values for Hikari datasource configuration. Separate keys and values with `=`. The valid keys can be found at HikariCP - GitHub.

Example to set the same as if selecting `recommended`:

```
maxLifetime=300000
idleTimeout=100000
maximumPoolSize=50
```

JavaObservability_deploymentEnvironment

Select a value for the OpenTelemetry `deployment.environment` attribute.

Choose between:

- `production`: example value used in OpenTelemetry documentation

- `staging` : example value used in OpenTelemetry documentation
- `testing`
- `development`

If nothing is selected, then this attribute will not be set.

In case the attribute is set in the `Agent Configuration` as well, the configuration provided here wins.

You may use this attribute for filtering, e.g. to separate information from prod and test for metrics and traces.

NevisIDMProperty_regex

Enter `regex` for the property definition file.

The defined regular expression will restrict the possible values that can be assigned to the property. If a value is entered, it will be checked against the specified pattern to ensure it meets the criteria.

Some examples of how regular expressions can be used for common data types:

Email address:

```
^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$
```

Telephone number in the format `+[country code] (XXX) XXX-XXXX` :

```
^\+\d{1,3}\s?\(\d{3}\)\s?\d{3}-\d{4}$
```

Social Insurance Number (SIN) in the format `XXX-XX-XXXX` :

```
^\d{3}-\d{2}-\d{4}$
```

URL in the format:

```
(https:\/\/www\.|http:\/\/www\.|https:\/\/|http:\/\/)?[a-zA-Z0-9]{2,}(\.[a-zA-Z0-9]{2,})(\.[a-zA-Z0-9]{2,})?
```

The regex will be escaped for JSON if required.

SecurityResponseHeaders_responseHeaders

Use this property to add security headers to responses. The syntax is: `<header name>:<value>`

Example:

```
Strict-Transport-Security: max-age=63072000
X-Content-Type-Options: nosniff
Referrer-Policy: strict-origin-when-cross-origin
```

CustomNevisMetaLogFile_serverLogFormat

Log4j 2 log format for the default SERVER logs.

Note: not relevant when Log Targets is set to `syslog`.

OAuth2Client_offlineAccess

Set to `allowed` to allow this client to request the scope `offline_access`.

When this scope is requested a refresh token will be returned as well.

CustomProxyLogFile_rotationType

Defines how to handle log retention for the access, apache and navajo log files. Rotation is possible based on:

- file size
- time interval

URLHandler_forwards

Rewrite the path of HTTP requests.

Rewrites are done using a `forward` which means that they are transparent for the caller.

The format is the same as in `Redirect Rules`.

NevisLogrendDeployable_path

Set a custom path for nevisLogrend resources (e.g. CSS). The path will be made accessible in nevisProxy.

You must change the path when using multiple nevisLogrend instances on the same virtual host.

TLSSettings_options

The value configured here will be applied as `SSLOptions`.

It should only have value when assigned to a `Virtual Host` pattern.

Check the Apache Documentation for details.

If empty and when this pattern is assigned to a `Virtual Host` the following value is used:

```
+OptRenegotiate +StdEnvVars +ExportCertData
```

NevisIDMSecondFactorSelection_notFound

Assign a step to continue with if the user does not have any supported credential.

Configuration is optional but we recommend to assign a step to handle the missing second-factor credential case. For instance, you may assign the following steps:

- `User Information` : to show an error message and terminate the authentication flow.
- `OATH Onboarding` : to register an authenticator app which supports OATH Time-based One-Time Password algorithm (TOTP).
- `FIDO2 Onboarding` : to register a FIDO2 authenticator such as a mobile device or USB security key.

AppleLogin_issuer

The `issuer` registered claim identifies the principal that issued the client secret. Since the client secret belongs to your developer team, use your 10-character Team ID associated with your developer account. Find out more here.

NevisDetectRiskPluginBase_dashboard

BehavioSec Dashboard URL

CookieCustomization_sharedCookies

Cookies listed here will be stored in nevisProxy and shared between all applications which have this pattern assigned.

Note that storing cookies requires a user session. Thus, we recommend to not use this feature for applications which are supposed to be stateless and public.

Regular expressions are supported.

Note that cookies matching `^Marker_.*$` will never be stored as a corresponding `allow` rule is generated to support `Session Expiration` features of the `SAML SP Realm`.

Example:

- `LANG.*`

OAuth2AuthorizationServer_preProcess

Assign a step to apply custom pre-processing logic.

This pre-processing logic is executed on the `Authorization Path` and `Token Path`.

You may assign a chain of steps to build a flow. The dispatching will continue when leaving this flow on the happy path.

For `On Success` exits this works automatically.

However, generic exits (i.e. `Additional Follow-up Steps` in `Generic Authentication Step`) must be marked a *success exits* by assigning the `Pre-Processing Done` pattern.

NevisDetectServiceAccessBase_token

Propagate a token to the backend application. The token informs the application about the authenticated user.

Please assign a `NEVIS SecToken`. This is mandatory to have access to the Administration UI.

NevisFIDODeployable_restrictAuthenticators

By default, all authenticators that fulfill the requirements given by the FIDO2 patterns are allowed.

Here you can restrict which authenticators are allowed based on metadata.

Select `enabled` here and provide the required metadata by configuring `Allowed Authenticators`.

OAuth2AuthorizationServer_transitions

Add or overwrite `ResultCond` elements in the `AuthorizationServer` state.

This setting is advanced. Use without proper know-how may lead to incorrect behavior.

If you use this setting, we recommend that you contact Nevis to discuss your use case.

The position refers to the list of `Additional Follow-up Steps`. The position starts at 1.

Examples:

`ResultCond`	Position
valid-token-request	1
valid-authorization-request	2

The following `ResultCond` elements cannot be overruled by this setting:

- `authenticate:valid-authorization-request`
- `stepup:valid-authorization-request`
- `server-error`
- `invalid-client` (configure `Invalid Client` instead)
- `invalid-redirect-uri` (configure `invalid Redirect URI` instead)

- `invalid-authorization-request` (configure `Invalid Authorization Request` instead)
- `invalid-token-request` (configure `Invalid Token Request` instead)

HeaderCustomization_responseHeaders

Adds/overwrites HTTP headers in responses.

The syntax is: `<header name>:<value>`

Force browser to use HTTPS only (1 day expiration):

```
Strict-Transport-Security: max-age=86400
```

Ensure pages are not cached:

```
Cache-Control: no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: 0
```

Headers set by Apache (e.g. `Server`) cannot be overwritten.

Note: change the `Filter Phase` to set headers early / late.

DeployableBase_instanceRenameDetection

During deployment nevisAdmin 4 checks if the instance has been renamed by checking the last metadata file deployed on the target host given the pattern ID.

If instance rename is detected the current instance is stopped.

This check should be disabled if multiple environments are simulated on the same server.

This setting is relevant for classic VM deployment only.

GenericNevisAuthSettings_javaOpts

Add additional entries to the JAVA_OPTS environment variable.

Use the expression `${instance}` for the instance name.

For instance, you may configure nevisAuth to create a heap dump on out of memory as follows:

```
-XX:+HeapDumpOnOutOfMemoryError
-XX:HeapDumpPath=/var/opt/nevisauth/${instance}/log/
```

Be aware that this example will not work for Kubernetes as the pod will be automatically restarted on out of memory and the created heap dump files will be lost.

JWTToken_algorithm

The following algorithms of JWT token are supported:

- `HS256` or `HS512` : compatible with `JWS` token type
- `RSA-OAEP-256` : compatible with `JWE` token type

CustomRiskScoreWeightConfiguration_reputationWeight

Configuration of the risk score weight for the ip reputation analyzer's risk score.

CustomAuthLogFile_auditSyslogFormat

Log4j 2 log format for the AUDIT SYS logs.

Note: not relevant when Log Targets is set to `default`.

SamlToken_type

- `Assertion` : produces a `SAML Assertion`.

Use for applications protected by Ninja.

- `Response` : produces a `SAML Response`.

Use for applications which have their own SAML SP. Assign the `SAML SP Binding` pattern to the application and link this pattern there.

Maintenance_path

Enter the base path under which the maintenance page will be hosted.

You usually don't have to change this configuration, unless the path clashes with any other hosted resources.

By default, `/maintenance/` is used.

NevisMetaDeployable_authSignerTrustStore

Assign a Trust Store which is used to validate the signature of a received NEVIS SecToken.

There are 2 use cases which require validation:

- when a user accesses the `nevisMeta Web Console` the SecToken is signed using `NEVIS SecToken / Key Store`.
- when nevisAuth calls nevisMeta the SecToken is signed using `nevisAuth Instance / Internal SecToken Signer`.

If no pattern is assigned the trust store will be provided by nevisAdmin 4 automatic key management. However, this requires that automatic key management is used in the `NEVIS SecToken` and `nevisAuth Instance` patterns.

SamlSpRealm_assertionConsumerService

Enter the path where SAML `Response` messages sent by the IDP shall be consumed.

This path also accepts `LogoutRequest` messages.

The IDP may send messages using POST or redirect binding.

NevisDetectDatabase_jdbcDriver

Due to licensing, nevisDetect cannot ship the JDBC driver to connect to Oracle databases, Therefore, those who want to use an Oracle database need to obtain and provide the Oracle JDBC driver on their own.

The `.jar` files can be downloaded from Oracle

Uploading any other `.jar` files containing JDBC drivers is possible as well.

NevisAuthRealm_logout

The default logout flow works as follows:

- the user accesses a protected application URL with query parameter `logout`, for instance `/myapp/?logout`
- any active session for this realm is terminated
- an HTML page is displayed that informs the user and allows to log in again on the same URL

To replace this behavior, assign a logout step here.

Note: to expose the logout function to the user, the web page(s) of backend applications should to be customized. Supported approaches are:

- Adapt the backend application: add a button or link that requests a URL ending with `?logout`.

Note that when this realm is used as a SAML IDP then the logout flow cannot be customized. The only thing you can do is to assign a `Logout` pattern and configure a custom `Redirect`. This is the location the user is redirect to after the SAML logout flow is done.

NevisAuthRealm_labels

Labels are used to show text in the language of the user.

Which labels are used depends on the assigned steps. Click `Download Default Labels` to retrieve the used labels and their translations.

Here you can overwrite the defaults and add your own translations or even add new labels, which may be required when using a `Custom Login Template` or `Generic Authentication Step`.

Upload 1 file per language code. The file name should be `labels_<code>.properties`. Check `Languages` on the `nevisAuth Instance` for enabled language codes.

The uploaded files **must** be UTF-8 encoded or special characters must be HTML encoded.

If you want to reuse existing `text_<code>.properties` and `LitDict_<code>.properties` files, you have to merge them first, or set `Translations Mode` to `separate`.

By default, the patterns add several default labels and the labels configured here are added on top. This is convenient as you only have to define labels that you want to add or overwrite. However, this way you cannot remove labels. If you want to do that you have to set `Default Translations` to `disabled` and then only the uploaded labels will be used.

The default login template uses the following labels:

- `title` - used as browser page title
- `language.<code>` - used by language switch component

The default logout process of nevisAuth (which will be applied when no step is assigned to `Logout`) has a confirmation GUI which uses the following labels:

- `logout.label` - header of the logout confirmation GUI
- `logout.text` - text shown to the user
- `continue.button.label` - label on the confirmation button

HostContext_sessionResourceAccessRestriction

Assign an access restriction patterns to prevent unauthorized access to the REST interface of the session resource.

8.2411.0

Full changelog:

Patterns 8.2411.0 Release Notes - 2024-11-20

NevisProxy Observability pattern refactor

The `NevisProxy Observability` pattern was refactored:

- Renamed the `Trace Resource Service Name` parameter and moved to the `Basic Settings` tab. It now controls the `service.name` key-value pair resource attribute for both the Metrics and Trace. If the default value was overwritten, the new parameter needs to be set.
- New configuration options have been added, such as `Sampler`, `Deployment Environment`, etc.
- Removed the experimental label from the pattern.

Maintenance Page pattern improvements

- The `Maintenance Page` pattern now includes its sanitized name in the names of the generated `MaintenanceFilter` and `DefaultServlet`. This prevents naming collisions, and allow linking several `Maintenance Page` patterns to a Virtual Host or an Application.
- The UpdateInterval is now configurable.

Check your configuration if you use a `Generic Application Settings` pattern or a `Generic Virtual Host Settings` pattern to customize your `MaintenanceFilter` or the related `DefaultServlet`.

Automatic migration

The `nevisadmin-plugin-nevisadapt` plugin was separated from the `nevisadmin-plugin-nevisdetect` plugin. As a result, the class names are updated accordingly.

`nevisAdapt Instance -> Observation timeframe` was moved to its own pattern (`nevisAdapt Observation Cleanup Configuration`). As a result (if `Observation timeframe` was not empty), the deprecated property has to be removed and the new pattern has to be generated for each instance to keep the configuration value.

DatabaseBase_trustStore

Assign a trust store which provides the CA certificate of the DB endpoint.

NevisIDMClient_clientExtId

External ID of the new client.

BackendServiceAccessBase_backends

Enter the complete URLs (scheme, host, port and path) of the backend services.

Note:

- all URLs must use the same scheme and path.
- automatic path rewriting will be performed when the path differs from the `Frontend Path`.

In case you are setting multiple addresses, use `Load Balancing` to select a request dispatching strategy.

NevisAdaptRememberMeConnectorStep_originalAuthenticationFlow

Set the first step of the full authentication flow to continue with in case no valid remember-me cookie was found:

- the remember-me cookie is not present in the headers
- the remember-me cookie is present but no longer valid
- the associated user is no longer active
- the browser fingerprint has changed

CAUTION: It will disable the remember-me functionality if you set it to the same step as the `On Success`.

NevisIDMJmsQueues_truststore

You should add a CA certificate, and then use a `PEM Trust Store` to provide it.

CustomRiskScoreWeightConfiguration_deviceWeight

Configuration of the risk score weight for the device cookie analyzer's risk score.

SamlSpIntegration_parameters

Define custom `init-params` for the nevisProxy `DelegationFilter` which propagates the SAML Response to the backend.

This setting is experimental and may be adapted in future releases.

Examples:

- `DelegatePostPolicy: override` - create a new POST request and send it to the `Assertion Consumer Service Path`. The response is returned to the client which means that the original (GET) request is lost. However, the SP can redirect to the application. This mode should be preferred for proper SP integration.
- `DelegatePostPolicy: sidecall` - send a POST request to the `Assertion Consumer Service Path` but do not return the response to the client. Afterwards, the original (GET) request is sent. This mode may be required in case the response of the POST request does not redirect to the application.

NevisIDMURLTicketConsume_idm

Assign a `nevisIDM Instance` or `nevisIDM Connector`.

OAuth2AuthorizationServer_properties

Configure properties of the nevisAuth `AuthorizationServer`.

Add or **overwrite** properties by entering a value.

Remove properties generated by this pattern by leaving the value empty.

Examples:

Key	Value
propagationScope	session
nevismeta.blockClientInterval	600

NevisFIDODeployable_addons

Assign add-on patterns to customize the configuration of nevisFIDO.

GenericSocialLogin_redirectURI

The callback URI to go to after a successful login with the social account.

This will create an endpoint in your host config.

The URL will be a combination of the `Frontend Address` of the `Virtual Host` and the value configured here. For example, let's assume that you have configured:

- Return Path: `/oidc/app/`
- Frontend Address: `https://nevis.net`

Then the URL will be `https://nevis.net/oidc/app/`.

Use the `exact:` prefix to use the given path as-is. Without this prefix a normal mapping with `/*` will be generated and thus sub-paths will be accessible as well.

SamlSpConnector_encryptionCert

Assign a pattern to configure the certificate to encrypt the outgoing message to the service provider.

NevisIDMChangePassword_lockWarn

Assign an authentication step to execute when the status of the URL ticket or credential is **lockWarn**.

GenericAuthRealm_authStatesFile

Upload an XML file containing `AuthState` elements.

Upload of a complete `esauth4.xml` is **not** supported.

The `Domain` element is optional.

- If missing the element will be created. The `Entry` methods `authenticate` and `stepup` will be set to the first provided `AuthState`. The method `logout` is not set and thus the nevisAuth default behaviour applies.
- If provided the `Domain` must come before all `AuthState` elements. The attributes `name` and `default` are not supported and should be omitted. Attributes are sorted by name. The `Entry` elements are sorted by `method`.

The `AuthState` linked to `stepup` should be able to dispatch the request. For instance, you may have assigned an `Authorization Policy` to your application(s) and thus you need a state which decides based on the request variable `requiredRoles`.

The following example dispatches level `2` into an `AuthState` named `TAN` which provides authentication via mTAN:

```
<AuthState name="EntryDispatcher" class="ch.nevis.esauth.auth.states.standard.ConditionalDispatcherState" final="false">
    <ResultCond name="nomatch" next="Authentication_Done"/>
    <ResultCond name="level2" next="TAN"/> <!-- TAN state is expecetd to set authLevel="2" -->
    <Response value="AUTH_ERROR">
        <Arg name="ch.nevis.isiweb4.response.status" value="403"/>
    </Response>
    <property name="condition:level2" value="${request:requiredRoles:^2.*$:true}"/>
</AuthState>
```

The following expressions are supported:

- `${instance}` : name of the nevisAuth instance
- `${request_url}` : generates a nevisAuth expression which returns the URL of the current request
- `${realm}` : name of the Realm (see below)
- `${keystore}` : name of the `KeyStore` element provided by this pattern. Assign a pattern to `Key Objects` to add a `KeyObject` into this `KeyStore` .

The `name` of `AuthState` elements is prefixed with the sanitized name of the Realm (referred to as `${realm}`).

The realm prefix must be added when using `propertyRef` to reference AuthStates generated by other patterns (e.g. `<propertyRef name="${realm}_SomeState"/>`).

An exception is the AuthState which defines the nevisIDM connection (as generated by `nevisIdm Password Login` or `nevisIDM Connector for Generic Authentication`). Here the `propertyRef` must be defined as follows:

```
<propertyRef name="nevisIDM_Connector"/>
```

This pattern does not validate that labels are translated. Translations can be provided on the `Authentication Realm` pattern.

OAuth2AuthorizationServer_realm

Assign a realm which shall be exposed as an OAuth2 Authorization Server or OpenID Connect Provider.

TCPSettings_connectTimeout

Timeout for establishing the TCP connection.

NevisProxyObservabilitySettings_resourceServiceName

Configure the `service.name` key-value pair resource attribute.

SamlSpConnector_sessionIndex

Set current session ID to SAML Response in `SessionIndex` element. This element is required for IDP-initiated logout and SP-initiated logout when Logout Type is set to `SOAP` .

The element will be included in SAML Response like:

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:Response Destination=... >
    ...
    <saml2:Assertion ID=...>
        ...
        <saml2:AuthnStatement SessionIndex="JeBKZSJah-0m2QjC4LJ8u74LUOY2ayAeenlPgBOx1N8" ... >
            ...
        </saml2:AuthnStatement>
    </saml2:Assertion>
</saml2p:Response>
```

And in SAML LogoutRequest, the SessionIndex element will be included like:

```
<?xml version="1.0" encoding="UTF-8"?>
<saml2p:LogoutRequest Destination=...>
    ...
    <saml2p:SessionIndex>JeBKZSJah-0m2QjC4LJ8u74LUOY2ayAeenlPgBOx1N8</saml2p:SessionIndex>
</saml2p:LogoutRequest>
```

GenericNevisLogrendSettings_javaOpts

Add additional entries to the JAVA_OPTS environment variable.

Use the expression `${instance}` for the instance name.

For instance, you may configure nevisLogrend to create a heap dump on out of memory as follows:

```
-XX:+HeapDumpOnOutOfMemoryError
-XX:HeapDumpPath=/var/opt/nevislogrend/${instance}/log/
```

Be aware that this example will not work for Kubernetes as the pod will be automatically restarted on out of memory and the created heap dump files will be lost.

LdapLogin_properties

Set custom properties for the UseridPasswordAuthenticateState.

Examples:

```
searchSizeLimit = 512
```

LogSettingsBase_logLevel

Change the level of the root logger. This impacts all logging apart from `Log Levels`.

Note that Syslog appenders have a threshold which ensures that only `INFO`, `WARN`, or `ERROR` messages are forwarded.

NevisAuthDeployable_sessionIndexingAttribute

Enter the name of a session variable for the session index.

FIDO2Onboarding_failedScreenButton

Configure to add a dispatcher button to the failed screen.

The button may have a special `Button Name` to render in a nice way by a customized `Login Template`.

For instance, Identity Cloud uses this mechanism to add a button which looks like a back arrow. This button takes the user to a previous step.

This is an advanced setting. Use only when you understand the concept.

GenericIngressSettings_caSecret

Enter the name of the Kubernetes secret which contains the CA certificate in the key `ca.crt`. If the secret does not exist it will result in `403 (Forbidden)`, and with a missing `ca.crt` key the feature will not be enabled.

Can be created with: `kubectl create secret generic ca-secret --from-file=ca.crt=ca.crt`

The `ca.crt` file can contain multiple certificates.

NevisIDMDeployable_encryptionFallback

Initialization vector ("iv") fallback mechanism

This must be set to true for customers who previously had no value set for `security.properties.key / propertiesKey` (old name) in the properties file *and* had encrypted values already stored in the database. Otherwise, the old values encrypted by default value will not be readable. If the database does not contain encrypted properties or unused URL tickets, it is safe to leave this turned off, and it is adviced to be turned off for stronger security.

NevisFIDODeployable_deepLinkAppHost

If you have uploaded any `Deep Link App Files`, then assign a `Virtual Host`.

The files will be hosted with a base path of `/.well-known/`.

The domain of the `Deep Link` must point to this `Virtual Host`.

If the user does not have the mobile app installed, the `Deep Link` will be opened in the browser instead.

NevisIDMCheckUserCredentials_credentialTypesToCheck

Credential types which existence for the user should be checked.

Possible values:

- `PASSWORD`
- `CERTIFICATE`
- `SECURID`
- `TICKET`

- SAFEWORDUSER
- OTP
- TEMPSTRONGPASSWORD
- GENERIC
- KERBEROS
- MTAN
- VASCO
- PUK
- URLTICKET
- DEVICEPASSWORD
- MOBILESIGNATURE
- SAMLFEDERATION
- SECURITYQUESTIONS
- CONTEXTPASSWORD
- OATH
- FIDO_UAF
- RECOVERY_CODE
- FIDO2

NevisMetaRESTServiceAccess_backendTrustStore

Assign the Trust Store provider for outbound TLS connections. If no pattern is assigned a trust store will be provided by the nevisAdmin 4 PKI.

LdapLogin_subtreeSearch

If `disabled` all the users to authenticate must be in the same directory node, specified in the properties `Base DN` and `User Attribute`. In this case nevisAuth uses the user's account to authenticate against the LDAP directory.

If `enabled` a search query for the user is performed, with the specified `Base DN`.

NevisFIDODeployable_username

Configure what is expected as `username` in incoming API calls.

Choose between `extId` and `loginId`.

NevisAuthConnector_url

Enter `hostname:port` of the nevisAuth instance.

UserInput_greeting

Enter a text or *litdict key* to be displayed in a line below the title.

The text should inform the user what has to be entered in this form.

NevisIDMChangePassword_clientNotFound

Assign an authentication step to execute when the status of the URL ticket or credential is **clientNotFound**.

NevisMetaConnector_url

Enter `hostname:port` of the nevisMeta instance.

NevisFIDODeployable_signerTrustStore

Assign the Trust Store provider for SecToken verification.

NevisDetectDatabase_user

Enter the user for the DB connection.

NevisAdaptFeedbackConfig_tokenLifetime

Set the maximum lifetime for the feedback token.

CustomProxyLogFile_serverLog

Select if only log file should be used or if statements should also be forwarded to syslog.

This property is relevant for classic VM deployments only. In Kubernetes the main logs are written to system out so that log messages appear in the docker logs.

Choose between:

- `default` - log to a file
- `default + syslog` - log to a file and forward to syslog
- `syslog` - forward to syslog only. The syslog facility is `localhost3` and the threshold is `INFO`.

SwissPhoneChannel_sender

The sender phone number to use to transmit the SMS.

SamlResponseConsumer_postProcess

Assign a step to apply custom post-processing logic, e.g. to enrich the authenticated user.

NevisIDMUserCreate_optionalAttributes

Define which attributes are optional and how to provide them.

Example:

```
firstName: ${sess:given_name}
name: ${sess:family_name}
country: ${sess:country}
```

NevisDetectDeployableBase_logging

Assign `nevisDetect Log Settings` to change the log configuration.

SamlSpConnector_outEncrypt

Select a part of the outgoing message going to be encrypted from the service provider.

AutomaticTrustStoreProvider_keystore

Assign one or multiple `Automatic Key Store` patterns to establish a trust relation.

GenericThirdPartyRealm_rolesFilter

Define the filter that shall be application to applications to enforce the presence of certain roles.

The following expressions may be used:

- `${realm.id}` - unique ID of this realm pattern
- `${realm.name}` - name of this realm pattern
- `${auth.servlet}` - name of the servlet of the `Authentication Application` . May be used to perform a side-call.
- `${filter.name}` - a proposed filter name calculated from the required roles
- `*{roles}` - duplicates the entire line once for each role

NevisIDMUserUpdate_allowOverwrite

If `enabled` , the attribute or property will be stored even when there already is a stored value.

If `disabled` , the stored value remains unchanged in this case.

CustomRiskScoreWeightConfiguration_countryWeight

Configuration of the risk score weight for the suspicious country analyzer's risk score.

OAuth2AuthorizationServer_host

Assign a `Virtual Host` which shall serve as entry point.

NevisProxyObservabilitySettings_sampler

Configures the available head sampling methods. Possible values are:

- **AlwaysOn**: Samples every trace. With high traffic in a production application it may cause significant overhead.
- **AlwaysOff**: Samples no traces. NevisProxy still generates the spanID for internal trace ID.
- **TraceIdRatio:**: Samples a given fraction of traces based on the configured `ratio` .
- **ParentBased:<delegate_sampler>**: Makes the decision based on the parent of the span. If the span has a parent, the sampler flag of the parent span will decide. If there is no parent span, the delegate sampler is used, that can be any of the samplers above.

SamlSpRealm_labels

Labels are used to provide human-readable text in the language of the user. Here you can overwrite the defaults and add your own translations.

The name of uploaded files must end with the language code. As the format is compatible you may upload existing `text_<code>.properties` files of nevisLogrend or `LitDict_<code>.properties` of nevisAuth.

The encoding of uploaded files does not matter as long as all translations are HTML encoded.

So far this property is relevant only if the `Logout Reminder` feature is enabled because then a page will be rendered. The following labels are used:

- `title` - used as browser page title
- `logout.text`

- `language.<code>` - used by language switch component
- `info.logout.reminder`
- `continue.button.label`

NevisProxyDatabase_path

Enter the path where the local session store shall be exposed and accessed by the nevisProxy peer.

GenericServiceSettings_removeFilterMappings

Remove `<filter-mapping>` elements generated by other patterns.

The syntax is a map of `<filter-name>:<url-pattern>` , according to elements from the `web.xml` .

In the `<filter-name>` the expressions `${service.name}` and `${realm.name}` may be used.

For applications which have only 1 frontend path you may use `${service.mapping}` instead of `<url-pattern>` .

Examples:

```
ModSecurity_${service.name}:${service.mapping}
Authentication_${realm.name}:${service.mapping}
```

NevisDetectDeployableBase_addons

Assign an add-on pattern to customize the configuration.

WebApplicationAccess_csrf

Cross-Site Request Forgery (*CSRF*) is an attack to force an authenticated user to send unwanted requests.

- `off (default)` - no CSRF protection. Recommended for applications which may be called from other sites.
- `header-based` - `GET` and `HEAD` requests are allowed (assumption: these methods must not manipulate server-side state). For other requests the `Referer` and `Origin` headers must match the `Host` header.

GenericSocialLogin_clientId

The identifier provided by the social account when you register with it as the IdP service.

AuthCloudBase_accessKeysJson

Upload the `access-keys.json` of your Authentication Cloud instance.

The file contains the instance name and an access key.

You can download this file from the NEVIS Authentication Cloud Console.

Check Integrate Authentication Cloud with nevisAdmin 4 for setup instructions.

NevisProxyDeployable_passwordGetter

Choose between:

- `recommended` : uses `nevisadmin` for Kubernetes and classic deployments. The recommended value may change in future releases.

- `nevisadmin` : uses a script deployed by nevisAdmin. Does not work for PKCS11.
- `nevisproxy` : uses `/opt/nevisproxy/bin/keystorepwget` to lookup passwords for encrypted key material. Requires nevisProxy version `4.4` or later. Does not support lookup of the password for the `key.pem` of `PEM Key Store` .
- `neviskeybox` : uses `/opt/neviskeybox/bin/keystorepwget` to lookup passwords for encrypted key material. nevisKeybox must be installed on the target system. Does not work in Kubernetes deployments.

NevisAdaptDeployable_database

Add a database connection reference pattern.

Required properties to be set in the connector pattern are as follows:

- JDBC Driver (Oracle or MariaDB)
- JDBC URL
- DB user/password

NevisIDMPasswordLogin_policyName

Enter the name of a nevisIDM URL Ticket policy to use for the URL Ticket that is created at the beginning of the password reset process.

If nothing is configured here the default URL Ticket policy will be used.

Among others, the policy defines how the link is communicated to the user (e.g. by sending an email) and sets the expiration.

You can create additional policies via the nevisIDM Admin GUI or via SOAP / REST API.

NevisIDMChangePassword_showGUI

Sets if the authState's GUI should be rendered, default is `enabled`.

If not set or set to `disabled`, the GUI will not be rendered, making `New Password` setting mandatory. `Current Password` and `New Password Confirmation` settings may also be required, depending on other settings.

PropertiesTestPattern_patternReferenceProperty

Reference one of the allowed patterns.

NevisFIDODeployable_port

Enter the port for the HTTPS endpoint.

NevisAdaptAuthenticationConnectorStep_profile

The profile used during processing the results of the analysis done by the nevisAdapt service.

There are 2 ways to react on the returned values:

- React on the returned events directly
- React based on the calculated weighted sum of the risk scores

Supported values are:

- `balanced` - balanced risk profile
- `strict` - strict risk profile with higher weights
- `custom` - to define own weights for the risk profile
- `events` - react on the returned events instead of the risk scores

You can find more information about the Risk profiles in the documentation.

GenericAuthenticationStep_entryState

Define the `name` of the first `AuthState`.

If not set the sanitized name of the pattern will be used.

The XML must contain an `AuthState` which has this name set, or one that uses the expression `${state.entry}` for the name.

CustomRiskScoreWeightConfiguration_velocityWeight

Configuration of the risk score weight for the ip velocity analyzer's risk score.

UserInformation_buttonType

Adds a button to the GUI:

- `none` - no button is added
- `submit` - adds a submit button. To continue with `On Submit` the `Message Type` must be `warning` or `info`.
- `cancel` - adds a cancel button. The button restarts the authentication flow from the beginning. The flow must be reentrant.

GenericServiceSettings_servlets

Configure `servlet` and/or `servlet-mapping` elements using the XML constructs described in the nevisProxy Technical Documentation.

You may **add** new elements or **customize** elements provided by other patterns.

- Reference a `servlet` by setting `servlet-name`. Use `Connector_${service.name}` for the servlet which connects to the backend application.
- Reference a `servlet-mapping` by setting `url-pattern`.

In Kubernetes side-by-side deployment a postfix is added to service names. Use the expression `${service.postfix}` connecting to a service deployed against the same inventory.

Example 1: Add or overwrite an `init-param`:

Enable load-balancing when there are multiple backend servers.

```
<servlet>
  <servlet-name>Connector_${service.name}</servlet-name>
  <init-param>
    <param-name>LoadBalancing</param-name>
    <param-value>true</param-value>
  </init-param>
</servlet>
```

Instruct nevisProxy to a add `Content-Type` header when missing.

```
<servlet>
  <servlet-name>Connector_${service.name}</servlet-name>
  <init-param>
    <param-name>ProxyPolicy</param-name>
    <param-value>mime-completion</param-value>
  </init-param>
</servlet>
```

Example 2: Remove an `init-param` (no `param-value` provided):

```
<servlet>
  <servlet-name>Connector_${service.name}</servlet-name>
  <init-param>
    <param-name>CookieManager</param-name>
  </init-param>
</servlet>
```

Example 3: Change the `servlet-mapping` for an application to use a different servlet by changing `servlet-name`.

```
<servlet>
   <servlet-name>Connector_Conditional_${service.name}</servlet-name>
   <servlet-class>ch::nevis::isiweb4::servlet::mapping::ServletMappingServlet</servlet-class>
   ...
</servlet>
<servlet-mapping>
   <servlet-name>Connector_Conditional_${service.name}</servlet-name>
   <url-pattern>${service.path}/*</url-pattern>
</servlet-mapping>
```

Removing `servlet` or `servlet-mapping` elements is not supported.

GenericAuthRealm_parameters

Define *Template Parameters*.

Examples:

```
smtp: smtp.siven.ch
sender: noreply@siven.ch
```

These parameters can be used in your `Configuration`.

The expression formats are:

`${param.<name>}`:

- `name` found: parameter value is used.
- `name` missing: expression is **not** replaced.

`${param.<name>:<default value>}`:

- `name` found: parameter value is used.
- `name` missing: default value will be used.

In `<default value>` the character `}` must be escaped as `\}`.

NevisProxyObservabilitySettings_metricsExporterAddress

Enter the target URL (`host:port`) of the backend services to which the exporter is going to send metrics. The `/v1/metrics` path is automatically attached to it.

NevisDPLogSettings_serverSyslogFormat

Log4j 2 log format for the SERVER SYS logs.

Note: not relevant when Log Targets is set to `default`.

SamlSpConnector_roles

Check for required roles.

Roles provided by nevisIDM have the following format: `<applicationName>.<roleName>` . Roles provided by other systems (e.g. LDAP) may have a different format.

Examples:

```
myApp.Admin
```

Required roles are always checked at the end of authentication, after enforcing the `Minimum Required Authentication Level` (optional).

The user must have any of the enter roles to continue.

If the user does not have any of these roles, the authentication will fail and a SAML `Response` with status `AuthnFailed` message will be returned to the SP.

If you want to do custom error handling (e.g. show a GUI to the user), assign a step to `On Forbidden` .

GroovyScriptStep_customSteps

Assign follow-up steps.

For each step a *transition* (called `ResultCond` in `esauth4.xml`) is added. The name of the transition depends on the position in the list.

For instance, if 2 steps are assigned the following *transitions* will be added:

- `exit.1`
- `exit.2`

The Groovy script may trigger a certain transition by calling the method `response.setResult` handing over the name of the transition.

Example:

```
response.setResult('exit.1')
```

SamlSpRealm_authParams

Add custom `init-param` elements to **each** `IdentityCreationFilter` generated by this pattern.

This pattern generates 2 `IdentityCreationFilter` elements:

1. `Authentication_<name>` : enforces authentication for applications.
2. `SAML_<name>` : provides the `Assertion Consumer Service` and `Session Upgrade Path`

If you want to patch only one of these filters consider using `Generic Application Settings` instead.

Note that the parameter `InterceptionRedirect` of the `SAML_<name>` filter is forced to `never` . If you configure `InterceptionRedirect` here it will be ignored for this filter as leads to message loss in SAML POST binding.

Examples:

- `BodyReadSize = 64000`

NevisIDMProperty_propertyName

Enter `name` for the property definition file.

Technical name of the property. The name has to be unique among the properties of the same scope and within the same client.

NevisFIDODeployable_link

Choose between:

- `Deep Link` : configures nevisFIDO to use a *deep link* (recommended).
- `Custom URI` : configures nevisFIDO to use a *custom URI link*.
- `undefined` : this pattern won't generate any link dispatcher configuration.

Deep links are harder to set up but more recommended as they offer a better user experience.

Nevis recommends using Custom URI links in mobile only scenarios only. The end-user is always expected to click a link on the same phone as the Access App is running on.

Note: the selected type here and the corresponding URI/URL must be communicated when Ordering an Access App.

Further information about deep links and custom URI links can be found in the related settings.

NevisAdaptAuthenticationConnectorStep_customRiskScoreWeightConfiguration

Custom risk score weight configuration for the calculation. Set the weights to be considered for each risk score analyzer.

Analyzer list:

- Suspicious country
- Device cookie
- Fingerprint
- IP
- IP location

- IP velocity
- IP reputation

ServiceMappingReport

This report lists the applications of this project, how they are protected by a realm, and which filters are applied.
{{#.}} {{#addressMappings}} {{#backendAddress}} {{/backendAddress}} {{^backendAddress}} {{/backendAddress}} {{/addressMappings}} {{/.}}

Frontend Address	Backend Address	Virtual Host	Application	Realm	Filter Chain	nevisProxy Instance	Deplo
{{frontendAddress}}	{{backendAddress}}	n/a	{{hostContextName}}	{{serviceName}}	{{realmName}}	{{{filters}}}	{{depl

JSONResponse_onContinue

This exit will be taken when `Response Type` is set to `AUTH_CONTINUE` and the next request is received.

NevisDetectAuthenticationConnectorStep_jmsClientKeyStore

Used when simple or mutual (2-way) HTTPs is configured. If no pattern is assigned here automatic key management will provide the key store.

CustomProxyLogFile_logLevel

Sets the base log level of nevisProxy.

The level will be applied to `BC.Tracer.ThresholdBase` as follows:

- `ERROR : 3`

- `NOTICE : 5`
- `INFO : 6`
- `DEBUG : 7`
- `DEBUG_HIGH : 9`
- `TRACE : 10`

Note that if you only change log levels nevisProxy won't be restarted during deployment. The new configuration will be activated within 60 seconds.

SamlSpRealm_spLogoutTarget

Enter a path or URL to redirect to when an SP-initiated SAML logout completes on this SP.

The redirect is performed only when `Logout Mode` is set to `redirect-target` .

AuthorizationPolicy_forbiddenRolesMode

The `Forbidden Roles Mode` defines which `Forbidden Roles` are set for the current paths.

When combining several `Authorization Policy` patterns for an application, this setting allow inheriting the `Forbidden Roles` from a more general pattern.

Choose one of:

- `self-contained` : The `Forbidden Roles` defined in this pattern are applied to the current paths. They override any `Forbidden Roles` set on parent paths. If no `Forbidden Roles` are set in the current pattern, no forbidden roles will be enforced for the current paths.
- `inherited` : The `Forbidden Roles` in this pattern is not used. Use this setting if you have another `Authorization Policy` pattern applied to a parent path to inherit the configuration from. For the `Forbidden Roles` to be inherited from a particular parent, this setting has to be set to `default (self-contained)` in the parent pattern (otherwise you may inherit a value from a grandparent).

JSONResponse_code

Enter an appropriate status code for the HTTP response. If not set the code will be set based on the selected `Response Type`:

- `AUTH_ERROR: 401`
- `AUTH_DONE: 200`

OutOfBandMobileAuthStepBase_level

Set an authentication level to apply when authentication is successful.

The level is relevant only if there are is an `Authorization Policy` assigned to applications.

NevisIDMUserLookup_clientInput

Enable this to allow the user to enter the name of the *Client* (tenant) when logging in to nevisIDM.

If `disabled`, the input field is not shown and the Client `Default` is used.

NevisFIDODeployable_deviceServiceTimeout

Defines the maximum time difference that is accepted between the time in the `creationTime` attribute in device service requests and the time when the server processes the request.

This value is close to the time drift that is accepted between the mobile device clock and the server clock. If the time drift is bigger than this value, the operation will fail.

The default value is 5 minutes. If no time unit is provided, seconds will be used.

NevisIDMChangePassword_tmpLocked

Assign an authentication step to execute when the status of the URL ticket or credential is **tmpLocked**.

JWTToken_kid

The `kid` (key ID) Header Parameter is a hint indicating which key was used to secure the JWS. This parameter allows originators to explicitly signal a change of key to recipients.

When used with a JWK, the `kid` value is used to match a JWK `kid` parameter value.

For reference, please consult RFC 7515.

SamlIdp_metadataServicePath

Enter a path where the *SAML Metadata Service* shall be exposed on the assigned `Virtual Host`.

FIDO2StepBase_fido

Assign a `nevisFIDO FIDO2 Instance`.

NevisIDMPasswordLogin_unitAttributes

Enter unit attributes to fetch from nevisIDM. Enter 1 attribute per line.

The following unit attributes are supported:

- extId, state, name
- displayName, displayAbbreviation, location, description, hname, localizedHname
- ctlCreDat, ctlCreUid, ctlModDat, ctlModUid

LdapLogin_onPasswordExpired

Assign a pattern which defines the step that is executed when the user must change his password.

If no pattern is assigned the next AuthState is `Authentication_Failed` which terminates the authentication process.

NevisAdaptDeployable_ipToLocationUpload

Provide a file attachment for the IP-to-Location service to use.

Please consider uploading the file manually if its size exceeds 20MB, then adjust the path `ipToLocationMappingFile` in *nevisadapt.properties* after deployment if needed.

With file upload, only the IP-Country database is supported, with fields listed as follows (CC is the 2-letter country code, no header row):

```
"IP range min (decimal)","IP range max (decimal)","CC","COUNTRY"
```

The file must adhere to the following formatting rules: all fields must be separated by commas and surrounded by double-quotes. The IP ranges should not intersect each other. File name must end with either .csv or .CSV.

If IP velocity analysis is required, it is handled through IP2LOCATION updates. No other provider is supported at this point. Please switch to either `DB5BIN` or `DB5LITEBIN`.

The IP-mapping file has to be updated regularly for the service to stay relevant.

Uploaded files are *not* updated by default.

We recommend setting up periodic update of IP geolocation and reputation mappings.

CSRFProtectionSettings_headerBased

CSRF protection can be obstructive for some cross-domain use cases (e.g. federation or providing a public REST API).

OAuth2Client_email

Set to `allowed` to allow this client to request the scope `email`.

This scope produces a claim `email`.

ErrorHandler_keepHeaderStatusCodes

By default, HTTP headers are dropped when an error code is handled.

This avoids *information leakage* but can lead to *session loss* in some cases.

For instance, the nevisProxy session will be lost when all of the following holds:

- this pattern is configured to handle code `502`.
- the application is unreachable (`502` is produced).
- the nevisProxy session cookie is *renegotiated* (`Set-Cookie` header is set).
- user refreshes the page after the error page is shown.

To overcome this limitation you may enter `502` here.

Note that we are investigating additional measures and may adapt this property in future releases.

NevisIDMPasswordLogin_finalRedirect

Where to redirect to once the password reset is successfully completed.

See the "Email Sent Redirect" property for more information about the possible values.

Note that in this case, `referrer` can be very useful as it will redirect the client straight to the page he initially wanted to access before he started the password forgotten process.

FrontendKerberosLogin_keyTabFilePath

Enter the path of the Kerberos keytab file.

The path must exist on the target host(s) of the `nevisAuth Instance`.

This configuration is ignored when keytab file(s) are uploaded via `Keytab File`.

In complex setups with multiple `Kerberos Realms` and/or `Frontend Addresses` you may want to enter multiple keytab file paths.

Dispatcher_defaultStep

Assign the step to continue with if no transition matches.

HostContext_additionalStatusCodes

Allow non-standard HTTP status codes.

The configuration of additional status codes is required, for example, when using WebDav (HTTP status code `207` is used by WebDav).

ServiceAccessBase_truststore

Optional setting for enabling trust to HTTPS backends.

For securing production environments:

- set `Backend Addresses` starting with `https://`
- assign a `Trust Store` pattern containing the certificates required for verifying the backend certificate
- set `Hostname Validation` to `enabled`

NevisAdaptRememberMeConnectorStep_onSuccess

Decides what to do if the remember-me token is present and valid. Leave empty for skipping to the end of the authentication flow immediately.

CAUTION: It will disable the remember-me functionality if you set it to the same step as the `Original Authentication Flow` .

GenericHostContextSettings_mimeMappings

Set or replace `mime-mapping` elements.

Examples:

```
<mime-mapping>
    <extension>svg</extension>
    <mime-type>image/svg+xml</mime-type>
</mime-mapping>
```

The `mime-mapping` elements affect the entire `Virtual Host` and are used use to determine the `Content-Type` for responses.

nevisProxy always sets a `Content-Type` header for static resources served by the `Virtual Host`.

Further, nevisProxy can add a `Content-Type` header for resources served by applications. To enable this advanced feature assign `Generic Application Settings` to the application and set the parameter `ProxyPolicy` to `mime-completion`.

ResponseRewritingSettings_responseBodyContentTypes

Enter regular expressions to match the `Content-Type` of responses. If the expression matches, the response body is rewritten.

CustomRiskScoreWeightConfiguration_ipWeight

Configuration of the risk score weight for the ip analyzer's risk score.

RESTServiceAccess_jsonValidation

Choose between:

- `enabled` - **all** requests which have a request body must be valid JSON.
- `log only` - similar to `enabled` but violations are not blocked, only logged.
- `content-type` - validation is performed only when the `Content-Type` header matches `application/json`.
- `disabled`

NevisAuthDeployable_clientAuth

Enable to enforce 2-way TLS on the nevisAuth HTTPs endpoint.

This means that callers (e.g. nevisProxy or technical clients accessing nevisAuth REST APIs) must present a client certificate.

The `Frontend Trust Store` must contain the issuing CA.

CustomAuthLogFile_maxFileSize

Maximum allowed file size (in bytes) before rolling over.

Suffixes "KB", "MB" and "GB" are allowed. 10KB = 10240 bytes, etc.

Note: not relevant when rotation type is `time` .

NevisIDMPasswordCreate_showPolicyViolations

If set to `enabled` then after failed credential creation displays violated policies.

NevisAdaptDeployableBase_keyStore

Used when simple or mutual (2-way) HTTPs is configured. If no pattern is assigned here automatic key management will provide the key store.

NevisMetaDeployable_port

Port the nevisMeta instance is listening on.

TLSSettings_ciphers

The value configured here will be applied as `SSLCipherSuite` .

Check the Apache Documentation for details.

If empty and when this pattern is assigned to a `Virtual Host` the following value is used:

```
ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-CHACHA20-POLY1305:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA256
```

If empty and when this pattern is assigned to an application, default `SSLCipherSuites` from nevisProxy are applied. Check the nevisProxy Technical Documentation for details.

NevisIDMConnectorAddon_genericAuthPatterns

Any generic Auth pattern that should have access to the generated `AuthState` .

7.2402.0

Full changelog:

Patterns 7.2402.0 Release Notes - 2024-02-21

Changes to init-params in nevisProxy

The latest nevisProxy release has breaking changes which affect several init-params.

Examples:

- `AutoRewrite` of `Http(s)ConnectorServlet`
- `RenewIdentification` of `IdentityCreationFilter`

You may have to adapt your configuration if you are using `Generic Virtual Host Settings` or `Generic Application Settings` to add or patch servlets or filters.

Check the release notes of nevisProxy for further information.

Removed Virtual Host Observability Settings pattern

The `Virtual Host Observability Settings` has been removed. Due to the refactoring of the OpenTelemetry integration in nevisProxy, the configuration now applies to the whole instance.

Modified TLS Encryption setting in NevisIDM pattern

Possible value `enabled` is removed and new more fine graded value of 'trust', 'verify-ca' and 'verify-full' are added.

NevisAdaptDeployable_ipToLocationFileSelector

Set a file code only if the provider is IP2LOCATION or MaxMind and also set the access token in that case.

Provide a file code that identifies the database file to be downloaded.

The supported values are:

- `upload` - no update mechanism will be in place for custom uploads by default. Must be *.csv/*.CSV, up to 20MB.
- `DB1BIN` - commercial version, IP-Country
- `DB1BINLITE` - free version, IP-Country
- `DB5BIN` - commercial version, IP-Country-City-GPS
- `DB5BINLITE` - free version, IP-Country-City-GPS
- `Geo2-City` - MaxMind GeoIP2 City Database
- `GeoLite2-City` - free version of the MaxMind GeoIP2 City Database

nevisAdapt doesn't provide any access token by default. They have to be generated after registration (in case of the commercial version, purchase).

You can find more information about the supported geolocation databases at the IP2LOCATION and MaxMind websites.

HostContext_qosConfiguration

nevisProxy uses the mod_qos module to ensure quality of service (QoS). Choose between:

- `off` : the module is disabled on this virtual host.
- `standard` : provides a default configuration which protects against common denial of service (DoS) attacks.
- `custom` : configure `Generic mod_qos Configuration` via `Additional Settings` .

NevisIDMPasswordLogin_unitProperties

Enter unit properties to fetch from nevisIDM. Enter 1 property per line.

The properties must have scope onUnitGlobal. The property name must be exactly as defined in nevisIDM. Otherwise, the property value will never be written into the session.

CustomInputField_label

Enter a text or *litdict key* to be displayed as *label* in front of the input field.

MicrosoftLogin_clientExtId

The ExtId of the client in nevisIDM that will be used to store the user

NevisFIDODeployable_authenticationTimeout

Defines the maximum time duration between the generation of the `AuthenticationRequest` by nevisFIDO and the `AuthenticationResponse` by the FIDO UAF client.

If the client has not sent the response after this time, a client timeout occurs.

The default value is 2 minutes. If no time unit is provided, seconds will be used.

This timeout is relevant in the authentication use-cases, such as:

- In-Band Authentication
- Out-of-Band Authentication

NevisAdaptDatabase_hikariType

Select which method of generation should be applied when configuring the Hikari datasource for the database connection.

Possible options:

- `recommended` : the default option, this sets up three explicit values:
  - Maximum session lifetime: 300s
  - Session idle timeout: 100s
  - Maximum pool size: 50
- `custom` : specify values in the next text area, separate keys and values with `=` .
  - The valid keys can be found at HikariCP - GitHub.
- `unmodified` : this configuration doesn't generate anything, leaving all default configurations coming from the library in effect.

NevisAdaptRememberMeConnectorStep_adapt

Reference for the nevisAdapt service to check for the presence of the provided remember-me token.

SamlIdpConnector_assertionLifetime

SAML assertions have an issue timestamp. nevisAuth validates the timestamps of SAML assertions received from the IDP.

Some identity providers create the SAML assertion on login and return the same assertion as long as the session is active on the identity provider.

In this case enter a duration which is at least as long as the maximum session lifetime on the identity provider.

For identity providers which always return a new assertion (e.g. nevisAuth) the value can be very low (e.g. `30s`)

Enter `unlimited` to disable the maximum lifetime check for received SAML `Responses`. This sets `in.max_age` to `-1` in the generated `ServiceProviderState`.

OAuth2AuthorizationServer_clientExtId

The `extId` of the client in nevisIDM where the user is stored.

FacebookLogin_redirectURI

The callback URI to go to after a successful login with Facebook.

This will create an endpoint in your host config.

The URL will be a combination of the `Frontend Address` of the `Virtual Host` and the value configured here. For example, let's assume that you have configured:

- Return Path: `/oidc/facebook/`
- Frontend Address: `https://nevis.net`

Then the URL will be `https://nevis.net/oidc/facebook/` .

Use the `exact:` prefix to use the given path as-is. Without this prefix a normal mapping with `/*` will be generated and thus sub-paths will be accessible as well.

NevisIDMStepBase_nevisIDM

Assign a `nevisIDM Instance` or `nevisIDM Connector` .

NevisFIDODeployable_userVerificationTimeout

Maximum time that a FIDO2 client has to send the response in a ceremony where user-verification is required.

Default value is 5 minutes.

NevisDetectDatabase_encryption

Enables TLS in a specific mode. The following values are supported:

- `disabled` : Do not use TLS (default)
- `trust` : Only use TLS for encryption. Do not perform certificate or hostname verification. This mode is not recommended for production applications but still safer than `disabled` .
- `verify-ca` : Use TLS for encryption and perform certificates verification, but do not perform hostname verification.
- `verify-full` : Use TLS for encryption, certificate verification, and hostname verification.

AppleLogin_claimsRequest

The claims request parameter. This value is expected to be formatted in JSON and does not accept trailing spaces nor tabs.

StaticContentCache_maxLifetime

The maximum duration to cache a document.

NevisAdaptDeployable_ipToLocationHostnameVerifier

Enabling this option will set an Apache hostname verifier (which also handles certificate checks) instead of the default one.

Default: `disabled` (backwards compatibility)

NevisProxyDeployable_logging

Add logging configuration for nevisProxy.

GroovyScriptStep_validation

Choose between:

- `enabled` - parse the Groovy script and run against mock objects.
- `parse-only` - only parse the Groovy script.
- `disabled` - the script is not validated.

The validation is not feature complete and thus there may false negatives.

For instance, `import` statements can make the validation fail as the corresponding classes are usually not on the nevisAdmin 4 classpath. This case is quite common and thus failed imports will be reported as info issues to not block deployment.

If your Groovy script produces warning or error issues but is working inside nevisAuth please select `disabled` and provide the script to Nevis Security so that we can improve the validation.

When set to `enabled` the following mock objects will be used for validation:

```
Map<String, String> parameters
Map<String, Object> inctx
Properties inargs
Map<String, Object> session
Properties outargs
Properties notes
Request request
Response response
Tracer LOG
```

SamlIdpConnector_logoutType

Setting for SP-initiated logout, logout methods can be chosen

- **IMPLIED**: the logout method will be used by getting configuration of `Binding: Outbound`
- **POST**: force logout method to POST
- **SOAP**: force logout method to SOAP. This only work when IdP SAML Response contain `SessionIndex` .

InitRuntimeConfiguration_realm

Authentication realm to create the init runtime configuration.

CustomNevisIDMLogFile_auditSyslogFormat

Log4j 2 log format for the SERVER SYS logs.

Note: not relevant when Log Targets is set to `default`.

NevisIDMSecondFactorSelection_fido2

Assign a step which may be selected when the user has a FIDO2 Authenticator credential.

Assign a `FIDO2 Authentication` pattern here.

SAPLogonTicket_includeCertificate

When `enabled`, the signer's certificate is inserted into the issued SAP ticket.

NevisAdaptUserNotification_async

This property defines whether the communication should happen immediately (disabled) or via the EventQueue (enabled).

DatabaseBase_database

Here you can change the name of the database.

The database name **only** needs to be changed when the database service contains multiple databases.

AuthCloudOnboard_deepLinkLabel

Label to display on the element which allows the user to use the deep link to onboard.

The element is usually a button.

NevisFIDOServiceAccessBase_nevisfido

Assign a `nevisFIDO Instance` or `nevisFIDO Connector`.

GroovyScriptStep_scriptFile

Upload the Groovy script as a file.

Further information can be found in the nevisAuth Technical Documentation:

- ScriptState
- Writing scripts in Groovy

Use the expression `${service.postfix}` to refer to Kubernetes services deployed by this nevisAdmin 4 project.

The expression can always be used as it produces an empty String when the deployment is not a Kubernetes side-by-side deployment.

For instance, the following snippet declares a URL which points to the REST API of a `nevisIDM Instance` that has been deployed as a Kubernetes service called `idm`:

```
def url = "https://idm${service.postfix}:8989/nevisidm/api"
```

You may use `var` expressions to insert values from inventory variables at generation time. For instance, use `${var.<name>}` to insert a variable called `<name>`.

If the variable is a scalar, the value will be returned as-is. If the variable is a sequence, a Groovy list will be returned (start: `[`, end: `]`, separator: `,`, String quote: `"`).

If your Groovy script fails to validate, see `Script Validation`.

OATHAuthentication_level

Authentication level that is set on success.

NevisIDMSecondFactorSelection_otp

Assign a step which may be selected when the user has an OTP card credential.

Note that OTP card credentials may be used for various authentication methods (e.g. a one-time password list or VASCO Digipass devices).

NevisFIDODeployable_authenticationTokenTimeout

Defines the maximum time a client has to redeem an authentication token after the generation of the token by nevisFIDO.

Once the token is redeemed, the `Authentication Response Timeout` applies: the client has a maximum time to send an `AuthenticationResponse` to nevisFIDO.

This timeout is relevant in Out-of-Band Authentication.

NevisMetaDatabase_schemaPassword

The password of the user on behalf of the schema will be created in the database.

NevisIDMChangePassword_reenterOldPassword

If `enabled`, the user has to re-enter the old password before changing it.

If `disabled`, the user can change the password without re-entering the old password.

SamlIdpConnector_attributes

Configure to extract attributes from SAML assertions and store them in a session variable.

Examples:

Session Variable	Attribute
sess:user.email	email
sess:user.mobile	mobile

ServiceAccessBase_keystore

Optional setting to use a client certificate for connecting to HTTPS backends.

FIDO2Authentication_authStateClass

Select one of the available implementations.

When `ScriptState` is selected, all requests sent by JavaScript are directed towards nevisAuth. The script takes care of the communication with the nevisFIDO component, and thus you can restrict access to nevisFIDO. There is no need to expose any nevisFIDO APIs on the nevisProxy `Virtual Host`.

When `Fido2AuthState` is selected, configuration for `Fido2AuthState` is generated. FIDO2 related requests are sent to nevisFIDO instead. This requires that the following nevisFIDO APIs are exposed on the nevisProxy `Virtual Host`:

- `/nevisfido/fido2/attestation/options`
- `/nevisfido/fido2/assertion/result`
- `/nevisfido/fido2/status`

The easiest way to ensure this is to add a `nevisFIDO FIDO2 REST Service` pattern to your project.

It is recommended to select the `Fido2AuthState` implementation as it is a more pragmatic solution whereas the `ScriptState` is likely to be decommissioned.

This pattern is experimental and likely to change in future releases.

NevisLogrendDeployable_bindHost

The host name that this nevisLogrend instance will bind to. nevisProxy will connect to the same host name.

SamlSpRealm_issuer

Set the `Issuer` used by this SAML Service Provider.

The issuer can be an arbitrary string but it is recommended to use the complete URL that the *Assertion Consumer Service* is exposed on.

Example: `https://sp.siven.ch/SAML2/ACS/`