SamlIdp_metadataService

When `enabled` a *SAML Metadata Service* will be generated which can be accessed on the `Metadata Service Path`.

The SAML Metadata Service is not protected by authentication.

NevisDetectCoreDeployable_riskPlugins

List of Risk Plugins that are loaded by this nevisDetect Core component

Button_label

Enter the label which shall be used for this button.

The label should be added to the `Translations` of the realm.

TransactionConfirmation_host

Assign a `Virtual Host` to expose the transaction confirmation services.

The following public endpoints can be invoked:

- `/nevisfido/token/dispatch/authentication`
- `/nevisfido/status`
- `/nevisfido/token/redeem/authentication`
- `/nevisfido/uaf/1.1/facets`
- `/nevisfido/uaf/1.1/authentication`

4.15.0

Full changelog:

Patterns 4.15.0 Release Notes - 2022-05-18

nevisProxy Session Store

The nevisProxy `Virtual Host` now generates only 1 `servlet` for storing sessions. The servlets have fixed names:

- remote session store: `MySQLSessionStoreServlet`
- local session store: `LocalSessionStoreServlet`

If you are using `Generic Virtual Host Settings` to patch a session store servlet you will have to adapt the pattern configuration to use the new names.

Authorization Policy Pattern

Several new features have been added to the `Authorization Policy` pattern. The generated filters have been renamed.

If you are using `Generic Virtual Host Settings` to patch any of these filters, you will have to adapt the pattern configuration to use the new names.

Changes related to Log Settings

Several Nevis components have migrated from Log4J v1 to Log4J v2. The relevant Log Settings pattern have been adapted and aligned. Generic Log Settings patterns have been removed. Further, log settings for nevisFIDO have been moved into a separate pattern. If you have configured log settings you may have to adapt your pattern configuration. The issues generated during background validation will guide you through this process.

SamlSpConnector_issuer

Configure the *issuer* used by the SAML service provider.

NevisProxyObservabilitySettings_captureRespHeaders

HTTP client instrumentation will capture HTTP response header values for all configured header names.

Default in nevisProxy:

```
Content-Type, Content-Length, Content-Encoding, Location, Server, Connection, Keep-Alive, X-Forwarded-For
```

DeployableBase_openTelemetry

OpenTelemetry is used for several use cases:

- cross-component tracing in logs
- exposing metrics

By default, OpenTelemetry is `enabled` and a Java agent is loaded.

If that Java agent is not present on the machines you are deploying to, then you have to provide it at `/opt/agent/opentelemetry-javaagent.jar` or select `disabled`.

PropertiesTestPattern_bytesProperty

Enter a value in bytes, kilobytes, megabytes, gigabites. The unit should be entered with "k", "m", "gb". Ideally, we should have a widget which helps with the unit.

NevisMetaDeployable_clientAuth

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

NevisIDMPasswordCreate_onSuccess

Assign a step to continue with after successfully creating the password credential.

RealmBase_cookieName

By default, the session cookie will be called `Session_<pattern-name>`

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

If the same name is configured for multiple realms on the same host then the sessions will be cleaned up together when the first session expires.

OAuth2RegistrationEndpoint_path

Enter the URL of the registration endpoint.

Note that this pattern does **not** set up a registration endpoint, it just provides information about that endpoint.

The information is then used by the `OAuth 2.0 / OpenID Connect Metadata Endpoint` to provide metadata.

The prefix `exact:` is not supported here, enter the path as-is.

NevisDetectPersistencyDeployable_jms

Add reference for the pattern providing Java Messaging Service.

Two different options are allowed at this time:

- `nevisDetect Message Queue Instance` - deployment pattern for a dedicated MQ component
- `ActiveMQ Client Configuration` - connect to an external ActiveMQ service via SSL

WARNING: In case of Kubernetes deployment, only `ActiveMQ Client Configuration` is supported.

LdapLogin_connectionUsername

User to connect with. This user is part of the LDAP connection url.

Example:

- CN=admin,O=company,C=ch

NevisLogrendLogSettings_maxBackupIndex

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

GroovyScriptStep_parameters

Set parameters for your Groovy script.

Enter the **name** of the parameter as `Key`.

The `Value` can be either:

1. constant String value
2. nevisAuth expression (`${...:...}`)
3. an EL expression (`#{...}`)
4. a reference to an inventory variable (`${var.<name>}`). Such expressions are resolved during generation.

Parameters can then be used inside the Groovy script via the `parameters` map.

Example usage:

```
parameters.get('backend-url')
```

AuthCloudBase_instance

Instead of uploading an `access-key.json`, you can enter the name of your Authentication Cloud instance here.

NevisAuthDeployable_sessionLimit

Defines the maximum number of user sessions than may be created in this nevisAuth instance.

A nevisAuth session requires at least 10kb but the session can be much bigger when a user has many roles or multiple tokens are used.

SecToken_header

Set a custom header instead of the default `Authorization` header.

SAPLogonTicket_applicationMappings

A list of user ID mappings of the form `<application>:<ID>` to be inserted in the ticket. This will be used by SAP services to retrieve local user IDs. SAP NetWeaver Portal CRM plays a special role here as its user management is based on UME and, typically, has distinct IDs.

GenericModQosConfiguration_hostDirectives

Host level directives can be entered here.

BackendServiceAccessBase_hostHeader

Defines the `Host` header for requests forwarded to the application.

When `backend` is selected then nevisProxy uses the host part of the backend address that has been selected. This is the default behaviour and similar to what a browser would do. Therefore, this configuration should work in most cases.

When `client` is selected then nevisProxy will keep the `Host` header as received from the client. The following `init-param` will be generated:

```
<init-param>
  <param-name>HostName</param-name>
  <param-value>ENV:HTTP_Host;</param-value>
</init-param>
```

The configuration is dynamic to support virtual hosts with multiple frontend addresses. Note that this may be less secure. Even though browsers do not allow this clients may sent an arbitrary value for the `Host` header. It is therefore recommended to test how your application behaves in this case.

InBandMobileDeviceRegistration_authenticationService

Convenience feature.

If `enabled`, an endpoint will be provided at the `Authentication Service Path`.

The mobile app may use this endpoint to authenticate and obtain a cookie.

With this cookie the registration operation can be initiated.

The flow is described here.

There are several alternatives:

- use `Standalone Authentication Flow` to provide an authentication endpoint for this realm using authentication steps.
- authenticate the registration operation using the `Initial Authentication Flow` of the assigned `Authentication Realm`.

AccessRestriction_ips

List of client source IPs which shall be allowed. You may include entire range of IPs by separating two IPs with `-`. If there is load-balancer in front of nevisProxy please configure it to preserve the client source IP. IPv6 is not supported here.

Examples:

- `10.0.0.1` : specific IP address
- `192.168.0.0-192.168.0.255` : range of IP addresses

- `0.0.0.0-255.255.255.255` : all IP addresses

NevisFIDODeployable_firebaseServiceAccount

For sending push notifications, nevisFIDO needs to access Firebase, which is a push messaging service. For that, it requires an account and its corresponding credential.

Please visit the Firebase Console, create a project and download the `service-account.json` file. Please upload this file here.

Note that this file contains a private key, that gives access to your project's Firebase services. Keep it confidential at all times, and never store it in a public repository. Be aware that anybody who has access to this property, also has access to the file itself.

NevisAuthRadiusResponse_condition

An expression which defines when this response is generated.

For instance, use `${response:status:0}` to return this Radius response for all `AUTH_CONTINUE` responses. Likewise, you can use `1` for `AUTH_DONE` and `2` for `AUTH_ERROR` responses.

In complex authentication flows consisting of multiple steps it can be tricky to find a good expression which matches for one step only. Please contact your integration partner if you need support.

NevisAdaptUserNotification_sendingMethod

This mandatory property defines the communication method. For the configuration and usage of these methods, refer to the nevisIDM reference guide.

NevisFIDODeployable_registrationTokenTimeout

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

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

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

This timeout is relevant in the Out-of-Band Registration use-case.

OAuth2RestEndpointBase_path

If you enter a **path** the REST service will be generated and exposed on the nevisProxy `Virtual Host` assigned to the `OAuth 2.0 Authorization Server / OpenID Provider`.

The prefix `exact:` is not supported here, enter the path as-is.

If you enter a **URL** no REST service will be generated. Use this variant if you want to use an external service.

Either way, the information will be used by the `OAuth 2.0 / OpenID Connect Metadata Endpoint` to provide metadata.

NevisAuthRealmBase_initialSessionTimeout

Define the idle timeout of the initial session. The user must complete the authentication within this time.

NevisAdaptDatabase_oracleApplicationRoleName

Name of the application role for the oracle database used for the Kubernetes migration. It's recommended to keep the default value unless the pattern is used with an existing database that has a different one.

MobileDeviceDeregistration_host

A virtual host assigned will be used to expose the protected services.

NevisAuthDatabase_schemaUser

The user which will be used to connect to the database and create the schema (tables).

The database must have been created already (`CREATE DATABASE`) and the user must have `CREATE` privileges for this database.

If not set, the database connection user will be used.

Example: `schema-user`

NevisIDMJmsQueues_expiry

NevisIDM JMS Queue to which Expiry messages should be sent.

Only accepts URIs starting with `amqp` , `amqps` or `Endpoint=sb` . Validates only URIs with `amqp` or `amqps` schemes.

Messages in Expiry Queue are those messages which validTo time has passed without successful receive action and without failing for other reason. For further reference check `NevisIdm Technical documentation > Configuration > Components > Provisioning module > Provisioning providers` .

NevisAdaptAnalyzerConfig_suspiciousCountryAnalyzer

Used to disable suspicious country analysis. Use with caution.

OutOfBandMobileAuthentication_numberMatching

Enable/disable number matching in case of push notifications. If `enabled`, a 4-digit number will be displayed on the screen that you have to enter on your mobile device.

By default, it is `disabled`.

For more information, see Number Matching.

NevisAuthRealm_resetAuthenticationCondition

In some setups it is required to adapt the `resetAuthenticationCondition` of the `Domain`. You can configure a nevisAuth or EL expression here.

If the expression evaluates to `true` then the authentication flow is reset and the request is dispatched from the beginning.

StaticContentCache_requestHeaderMode

Request headers can force an intermediate server to override its cache and answer with the response from the original server.

Choose one of:

- **comply** : Follow the `Cache-Control: no-cache` directives sent by the client.
- **ignore** : Answer with the stored response even if the client sent a `Cache-Control: no-cache` directive.

NevisProxyObservabilitySettings_captureReqHeaders

HTTP client instrumentation will capture HTTP request header values for all configured header names.

Default in nevisProxy:

```
Content-Type, Content-Length, User-Agent, Referer, Host, X-Forwarded-For
```

TransformVariablesStep_emptyValue

Defines how to set the variable when null or an empty String shall be stored.

Choose between:

- `skip-variable` : do not set the variable. The current value is preserved.
- `clear-variable` : sets an empty String.
- `remove-variable` : removes the variable.

NevisDetectEntrypointDeployable_persistency

Add reference for a nevisDetect Persistency Instance pattern.

NevisAdaptAuthenticationConnectorStep_mediumThreshold

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

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

NevisAuthRealm_langCookieName

Enter a name of the cookie that nevisLogrend issues to remember the language of the user.

The same name will also be used in nevisAuth to determine the language.

Note that the language cookie name is an instance global configuration in nevisAuth. Enter the same value for all realms associated with the same nevisAuth instance.

ObservabilityBase_type

Choose agent type:

- `OpenTelemetry` to integrate with self-hosted observability stack or with an OpenTelemetry compatible cloud provider.
- `Application Insights` to integrate with Azure Application Insights.

OutOfBandMobileRegistration_profileId

Enter a variable expression for the profile ID.

The default works when this step is a follow-up of `nevisIDM Password Login` or `nevisIDM User Lookup`.

ErrorHandler_overwriteStatusCodes

Overwrite certain HTTP status code(s) by returning with the defined status code instead.

If for an error code both **Blocked Status Code** and **Overwrite Status Code** is configured, the **Blocked Status Code** will take precedent.

Examples:

```
404,406-499 -> 401
405 -> 200
```

NevisAuthDatabase_attributes

Add or overwrite attributes of the `RemoteSessionStore` and `RemoteOutOfContextDataStore` XML elements.

Supported attributes are described in the Nevis documentation:

- RemoteSessionStore
- RemoteOutOfContextDataStore

If you want to set an attribute only on one of the 2 elements use the prefix `session:` or `oocd:` as illustrated below.

Examples:

Attribute	Value
`syncPullInitial`	`true`
`session:reaperThreads`	`5`
`session:storeUnauthenticatedSessions`	`false`
`oocd:reaperPeriod`	`120`

NevisIDMUserLookup_onSuccess

For security reasons `nevisIDM User Lookup` alone is not sufficient to authenticate the user.

The authentication flow should contain another step which checks credentials of the user and sets an `Authentication Level`.

Thus, it is required to assign a step here which will be executed after the user has been looked up from nevisIDM.

Examples:

- `Authentication Cloud`
- `Mobile TAN (mTAN)`
- `Generic Authentication Step`

NevisAdaptAuthenticationConnectorStep_events

Will be considered only if `Profile` is set to `events`.

Select which events to react on. The events are identified and returned by the nevisAdapt service and the first event combination that they match successfully will determine the next step in the authentication flow. No further entries of this list will be considered.

One event combination entry consists of the following properties:

- `Risk Events` : set of suspicious event(s) to match against
- `Minimum Match Count` : minimum number of events to consider the matching valid (`all` by default). They have to be present in the service response to classify the entire combination as matching.
- `Authentication Step` : next authentication step if the matching is valid

Complete example with full ruleset:

Combination 1:

- Risk Events: ['ip-reputation-blacklisted', 'suspicious-country']

- Minimum Match Count: 1
- Authentication Step: Authentication Fails

This combination will match successfully if any of the two selected events are being reported by the nevisAdapt service. If this is the case, neither Combination 2 or 3 will be checked as the authentication fails immediately.

Combination 2:

- Risk Events: ['unknown-device', 'unknown-country', 'unknown-fingerprint']
- Minimum Match Count: 2
- Authentication Step: mTAN

This combination will match successfully if any 2 of the three selected events are being reported by the nevisAdapt service. If this is the case, Combination 3 will not be checked and the next authentication step will be mTAN.

Combination 3:

- Risk Events: ['unknown-country', 'high-ip-velocity']
- Minimum Match Count: all
- Authentication Step: email

This combination will match successfully only if both events were reported by the nevisAdapt service. If this is the case, a notification email will be sent to the user.

Otherwise, authentication succeeds without any further complication.

NevisMetaServiceAccessBase_backendTrustStore

Assign a trust store if you want to validate the server certificate used by nevisMeta. If this not set, the connection is 1-way TLS

GenericAuthRestService_configFile

As an alternative to direct configuration you can upload a file which contains the XML.

The file should contain `RESTService` elements only.

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

NevisAdaptObservationCleanupConfig_untrustedTimeframeDays

nevisAdapt stores session data that was not marked as trusted (e.g.: failed 2FA authentication) for a certain amount of time. This is done to allow staff to investigate the issue and to provide the user with a better experience. However, storing untrusted session data for too long can lead to privacy issues. This pattern describes how to configure the cleanup of untrusted session data.

The default value is `12d` .

NevisDetectCoreDeployable_jms

Add reference for the pattern providing Java Messaging Service.

Two different options are allowed at this time:

- `nevisDetect Message Queue Instance` - deployment pattern for a dedicated MQ component
- `ActiveMQ Client Configuration` - connect to an external ActiveMQ service via SSL

WARNING: In case of Kubernetes deployment, only `ActiveMQ Client Configuration` is supported.

RealmBase_sessionValidation

A newline separated list of rules declaring attributes that must not change in the same session. A rule has the following syntax:

```
ENV|CONST|PARAM|HEADER:<name of the attribute>:block|invalidate
```

- `block` : the request will be blocked and `403 (Forbidden)` will be returned
- `invalidate` : the session will be invalidated and a new one will be created

nevisProxy Conditions are supported. See nevisProxy reference guide for details.

For instance, use the following configuration to terminate the session if the source IP changes:

```
ENV:REMOTE_ADDR:invalidate
```

DefaultService_path

The path(s) which shall be accessible on the assigned `Virtual Host(s)` .

UserInput_title

Enter a text or *litdict key* for the form title (`<h1>`).

AuthCloudLogin_deepLinkLabel

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

The element is usually a button.

NevisAdaptFeedbackConfig_auth

Add nevisAuth Instance reference pattern(s) to enable session termination in connected components. If the session store is shared, it is enough to add one instance per database.

Please make sure that all involved nevisAuth Instances have ManagementService enabled. Add or extend a `Generic nevisAuth REST Service` for each with the following configuration:

```
<RESTService name="ManagementService" class="ch.nevis.esauth.rest.service.session.ManagementService" />
```

NevisAuthDeployable_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.

URLHandler_subPaths

Set to apply this pattern on some sub-paths only.

Sub-paths must be relative (e.g. not starting with `/`) and will be appended to the frontend path(s) of the virtual host (`/`) or applications this pattern is assigned to.

Sub-paths ending with `/` are treated as a prefix, otherwise an exact filter-mapping will be created.

The following table provides examples to illustrate the behaviour:

Frontend Path	Sub-Path	Effective Filter Mapping
/	secure/	/secure/*
/	accounts	/accounts
/	api/secure/	/api/secure/*
/	api/accounts	/api/accounts
/app/	secure/	/app/secure/*
/app/	accounts	/app/accounts
/app/	api/secure/	/app/api/secure/*
/app/	api/accounts	/app/api/accounts

GenericDeployment_owner

Owner of the directory at specified path. All files and subdirectories will have the same owner.

SocialLoginDone_status

Choose how to complete the flow:

- `AUTH_DONE` : user is authenticated
- `AUTH_ERROR` : session is terminated

In both cases the caller is redirect back to the path before jumping of to the social login provider.

When social login is behind federation (e.g. `SAML IDP`), `AUTH_ERROR` will be handled by sending the caller back to the origin (e.g. the `SAML SP`) with a technical error message.

NevisAuthDeployable_propagateSession

Define the value of the `AuthEngine` attribute `propagateSession`.

- `enabled: true` is set which makes nevisAuth return the user's session to nevisProxy on `AUTH_DONE`.
- `disabled: false` is set - nevisAuth does not return the session.

It is generally recommended to disable this feature and thus we plan to change the default to `disabled` in a future release.

DatabaseBase_parameters

Enter parameters for the DB connection string.

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

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

Enter 1 parameter per line.

Lines will be joined with `&`.

Examples (from various Nevis components):

```
pinGlobalTxToPhysicalConnection=1
useMysqlMetadata=true
autocommit=0
```

CustomProxyLogFile_rotationTime

Interval on which a logfile will be rotated.

NevisIDMSecondFactorSelection_recovery

Assign a step which may be selected when the user has a recovery codes credential.

For instance, assign a `Generic Authentication Step` pattern.

RealmBase_maxSessionLifetime

Define the maximum lifetime of a nevisProxy session. The session will be removed after that time even if active.

CustomNevisIDMLogFile_applicationLogFormat

Log4j 2 log format for the default SERVER logs.

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

TCPSettings_dnsCacheTTL

If `DNS Caching` is set to `true` , `DNS Caching Timeout` specifies how long the DNS info should be cached (in seconds) before getting again the IP address.

NevisIDMPasswordLogin_encryption

Set to enable form encryption.

This feature is still experimental in nevisAdmin 4 and has been added to this pattern as a preview.

The default template includes the required JavaScript (e2eenc.js) to perform client-side encryption of the form values.

OAuth2AuthorizationServer_removeEmptyClaimsInToken

Defines if the empty claim(s) will appear in the Access Token and ID Token.

- enabled: the ID Token and Access Token will not include empty claim(s).
- disabled (default): the ID Token and Access Token may include empty claim(s).

WebApplicationAccess_requestValidation

- `off` - no request validation
- `standard` - uses ModSecurity OWASP Core Rule Set (CRS) with default paranoia level 1 - Basic security
- `custom` - configure `Request Validation Settings` via `Additional Settings`
- `log only` - uses `standard` in log only mode

SamlSpConnector_attributes

Add attributes to SAML assertions.

Values may be static, produced by a nevisAuth expression (`${...}`), or an EL expressions (`#{...}`). This table shows how to enter the configuration:

Attribute	Value
some_attribute	`${...}`
some_attribute	`#{...}`
some_attribute	`some_value`

Set the log level `Vars = DEBUG` and check the nevisAuth `esauth4sv.log` to find out which variables may are available.

For instance, if you have a `nevisIDM Second-Factor Selection` pattern in your authentication flow, you can use the expression `${sess:user.mobile}` to add a `mobile` attribute.

FIDO2Onboarding_authenticatorType

Describes the authenticators' attachment modalities.

Allowed values:

- `any` - does not set a specific value accepting the standard's default
- `platform` - indicates a platform authenticator, such as Windows Hello
- `cross-platform` - indicates a roaming authenticator, such as a security key

NevisIDMUserCreate_onSuccess

Define how to continue after user creation.

PropertiesTestPattern_urlProperty

Enter a URL. By default, only allows HTTP and HTTPS but this can be changed in patterns.

GenericAuthService_host

Assign a `Virtual Host` which shall serve as entry point for this authentication service.

SamlSpConnector_signerTrust

Configure the trust store used to validate incoming SAML messages (e.g. `AuthnRequest` , `LogoutRequest`) which are sent by this SP.

NevisAuthRealm_authenticate

The initial authentication flow starts with the authentication step assigned here. To create a multi-step flow, you can reference further steps from within the first assigned step.

The initial authentication flow is applied on first access, when the client does not have an authenticated session.

Every time a step within the flow executes successfully, the authentication level defined in that step is added to the authenticated session.

NevisAuthRealm_tokens

Tokens assigned here may be created after successful completion of the `Initial Authentication Flow` .

To produce and forward a token to an application backend, reference the same token from the application's `Additional Settings` property.

GenericAuthRealm_keyObjects

Assign patterns to add `KeyObject` elements to the `KeyStore` provided by this pattern.

NevisIDMPasswordLogin_properties

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

Properties must be created in the nevisIDM via SQL.

GenericSocialLogin_userInfoEndpoint

The user information endpoint of the OAuth2 server. It's required when `providerType` has the value `OAuth2` .

DatabaseBase_databaseManagement

The pattern can set up the database, and it's schema when deploying to Kubernetes.

The `complete` option, on top of handling the schema migration, will do the initial database preparation like creating the actual database or tablespace in case of oracle, as well as creating the required database users.

The `schema` option will skip the initial preparation and will only take care of the actual schema migration. This requires the schema owner and the application user credentials to be present in the root credential secret. The root user information can be omitted with this option.

You can select `disabled` here to opt out. In this case you have to create and migrate the database schema yourself.

This feature is set to `recommended` by default which aims for the most convenient solution based on the deployment type. In case of Kubernetes deployments, it uses `complete` . In a classical VM deployment, it will use `schema` if the pattern allows setting `Schema User` and `Schema`

`Password` , otherwise it's `disabled` .

OAuth2AuthorizationServer_idTokenLifetime

How long the ID token should be valid per default (can be overwritten by the setting of individual client). At most a few minutes are recommended.

SamlIdp_path

Define paths for the following cases.

- **SP-initiated authentication**

Service providers may send a parameter `SAMLRequest` containing an `AuthnRequest` (using POST or redirect binding) to request authentication. On successful authentication the IDP returns a SAML `Response` .

On entry an initial session will be created. The session may expire during authentication due to timeout.

When this happens an error page (name: `saml_dispatcher`) with title `title.saml.failed` and error message `error.saml.failed` will be rendered.

- **SP-initiated logout**

Service providers may send a `LogoutRequest` (POST or redirect binding) to logout from this IDP and other service providers.

- **IDP-initiated logout**

Applications may have a link pointing to the IDP to trigger a global logout.

This link may point to:

- `<path>/logout` : to show a logout confirmation page (GUI name: `saml_logout_confirm` , label: `info.logout.confirmation`)
- `<path>/?logout` : to skip the logout confirmation page.

If a `Referer` header has been sent by the browser, the logout confirmation page will have a `cancel` button which redirects to the referer. Note that if the SP is NEVIS you may have to adapt the `Security Response Headers` of the `Virtual Host` . By default, the header `Referrer-Policy: strict-origin-when-cross-origin` is set and this will prevent the path being sent so the `cancel` button will redirect to `/` .

During SAML logout the IDP renders a GUI named `saml_logout` with the following hidden fields:

- `saml.logoutURLs` : the URL of the SPs including `LogoutRequest` message as query parameter
- `saml.logoutURL` : the URL to redirect to after successful logout

The default nevisLogrend template contains Javascript to invoke all `saml.logoutURLs` and redirect to `saml.logoutURL` after all requests have been sent. This is a best effort operation which means that the JavaScript does not check if the logout was successful.

- **IDP-initiated authentication**

Requests to the base path without `SAMLRequest` will trigger IDP-initiated authentication.

In this case the following parameters must be sent:

- `Issuer` : as entered for a `SAML SP Connector`
- `RelayState` : this parameter is returned to the SAML SP together with the `Response`

StaticContentCache_maxEntrySize

The maximum size of a document to be cached. Larger documents are never cached.

ProxyPluginPattern_description

Add description(s) for this proxy plugin

CustomInputField_variable

Enter `<scope>:<name>` of the variable which shall be set.

The following scopes are supported:

- `inargs`
- `notes`
- `sess` or `session`

For instance, enter `notes:loginid` to prefill the login form which is produced by the `nevisIDM Password Login` pattern.

BehavioSecPluginPattern_url

Service URL used to connect to the BehavioSec service from the plugin. For example: `https://mycompany.behaviosec.com/BehavioSenseAPI/`

FIDO2Onboarding_onUnsupported

Assign a step to continue with when the browser does not support FIDO2 WebAuthn.

FIDO2Authentication_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 authentication.

Allowed values:

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

RequestValidationSettings_whitelistRules

Configure *whitelist modifications*.

As explained in the ModSecurity documentation *whitelist modifications* are applied **before** including the core rules.

Note that new rule may require a rule ID which has to be unique for this pattern. Use the range 1-99,999 as it is reserved for local (internal) use.

- Remove rule with ID `900200` for the path `/app/some.html` :

```
SecRule REQUEST_URI "@streq /app/some.html" "pass,nolog,id:1000,ctl:ruleRemoveById=200002"
```

AuthenticationConnectorStepBase_onFailure

Set the step to continue with in case of error. If nothing is set, the authentication fails.

OAuth2AuthorizationServer_refreshTokenRotation

Defines if a new Refresh Token is issued together with the Access Token on the Token Endpoint while exchanging a refresh token for a new access token (`grant_type=refresh_token`).

- enabled, a new Refresh Token is issued, the existing Refresh token is deleted.
- disabled, the existing Refresh token is returned and remains valid.

NevisFIDOConnector_frontendAddress

Enter the address of the `Virtual Host` where the services of this instance are exposed.

Enter the address without any path component.

Example:

```
https://example.com
```

The entered value is used to calculate:

- AppID
- *Dispatch payload*

The *dispatch payload* informs the mobile device where to access nevisFIDO for the following use cases:

- Out-of-band Registration
- Out-of-band Authentication

AuthenticationConnectorStepBase_cookieDomain

If unset, the cookie will not be scoped to subdomains. Set this value to a specific domain to include more than one hostname.

Example: The user wants to login through example.com

If `no value` is given, the cookie will be effective for requests with the following addresses:

- https://example.com/one/two/three...

If the value is actually set as `example.com`, the cookie will be effective for requests against subdomains as well:

- https://shopping.example.com/one...
- https://account.example.com/two...
- https://example.com/three...

CustomAuthLogFile_regexFilter

If set, messages for `esauth4sv.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.*
```

NevisAdaptLogSettings_levels

Configure log levels.

See nevisAdapt Reference Guide, chapter `Logging Configuration` for details.

Comprehensive logging guide is found here.

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.

The default configuration is:

```
AdaptModules-Generic = INFO
ch.nevis.nevisadapt.util.logging.OpTracer = DEBUG
```

Examples:

```
org.springframework.web.filter.CommonsRequestLoggingFilter=DEBUG
```

CustomRiskScoreWeightConfiguration_fingerprintWeight

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

SecurityConfigReport

EXPERIMENTAL FEATURE - REPORT CONTENT WILL CHANGE IN FUTURE RELEASES

This report provides a detailed overview of the security configuration for virtual hosts and the backend applications on these virtual hosts.
{{#hosts}} {{#.}}

Virtual Host: {{host_name}}

The following table shows the settings at the virtual host level. These settings apply to all applications.

{{#tls_settings}} {{/tls_settings}}

Topic	Pattern Setting	Configuration	Scope

QoS Configuration (mod_qos)	{{qos}}	{{#qos_element}}
		`navajo.xml:`
		`{{qos_element}}`
		{{/qos_element}} {{#qos_server}}
		`web.xml (server directives):`
		`{{qos_server}}`
		{{/qos_server}} {{#qos_host}}
		`web.xml (host directives):`
		`{{qos_host}}`
		{{/qos_host}}
			server directives:
			all applications on nevisProxy instance.
			
			host directives:
			all applications on virtual host.

Unsecure Connection	{{unsecure_connection}}	`navajo.xml:` `{{context_element}}`	All applications on virtual host.

Require Client Certificate	{{client_cert}}	{{#ssl_element}}
		`navajo.xml:`
		`{{ssl_element}}`
		{{/ssl_element}}
			All applications on virtual host.

Topic	Pattern Setting	Configuration	Scope
Security Configuration (ModSecurity)	{{rules}}	{{rules_version}}	Applications on virtual host which have enabled *Request Validation (ModSecurity)*.
Allowed HTTP Methods	{{allowed_methods}}	`navajo.xml:` `{{context_element}}`	All applications on virtual host.

Security Response Headers	{{{response_headers}}}	{{#response_headers_filter}}
		`web.xml:`
		`{{response_headers_filter}}`
		{{/response_headers_filter}}
			All applications on virtual host (unless replaced on application-level).

TLS Settings	{{{tls_settings}}}	{{{tls_settings_details}}}	All applications on virtual host.

{{#services}} {{#.}}

{{service_type}}: {{service_name}}

Settings at the application level can override or complement the virtual host settings. The **Effective Configuration** shows the final settings for each application.
{{#request_validation}} {{! rendered only for Web Application pattern }} {{/request_validation}} {{#csrf}} {{! only Web Application and REST Service have CSRF Protection property }} {{/csrf}} {{#json}} {{! only REST Service has JSON Validation property }} {{/json}}

{{#soap_schema_files}} {{! only SOAP Service has SOAP Schema Validation property }} {{/soap_schema_files}} {{#tls_settings}} {{/tls_settings}}

Topic	Pattern Setting	Effective Configuration

QoS Configuration (mod_qos)	-	{{#qos_element}}
		`navajo.xml:`
		`{{qos_element}}`
		{{/qos_element}} {{#qos_server}}
		`web.xml (server directives):`
		`{{qos_server}}`
		{{/qos_server}} {{#qos_host}}
		`web.xml (host directives):`
		`{{qos_host}}`
		{{/qos_host}}

Unsecure Connection	-	`navajo.xml:` `{{context_element}}`
Require Client Certificate	-	{{client_cert}}
Request Validation (ModSecurity)	{{{request_validation}}}	{{{request_validation_details}}}
Allowed HTTP Methods	{{#allowed_methods}}{{.}} {{/allowed_methods}}	`{{#allowed_methods_resolved}}{{.}} {{/allowed_methods_resolved}}`

Topic	Pattern Setting	Effective Configuration
CSRF Protection	{{{csrf}}}	{{{csrf_details}}}
JSON Validation	{{json}}	{{{json_details}}}
SOAP Schema Validation	{{#soap_schema_files}} {{.}} {{/soap_schema_files}}	{{{soap_schema_files_details}}}

Security Response Headers	{{{response_headers}}}	{{#response_headers_filter}}
		
		`web.xml:`
		`{{response_headers_filter}}`
		
		{{/response_headers_filter}}

TLS Settings	{{{tls_settings}}}	{{{tls_settings_details}}}

{{/.}} {{/services}} {{/.}} {{/hosts}}

OnDemandEntry_condition

Enter a custom nevisAuth or EL expression.

If set the `Authentication Level` will not be used.

The step assigned to `On Entry` will be executed when the expression evaluates to `true` .

NevisProxyObservabilitySettings_metricsInterval

Interval of the metrics reader to initiate metrics collection.

AuthCloudBase_proxy

If you have to go through a forward proxy for the outbound connection to firebase enter the hostname:port here.

At the moment only HTTP proxy is supported.

OAuth2Client_accessTokenLifetime

Enter a custom lifetime for the access token.

If not set the value of the `OAuth 2.0 Authorization Server / OpenID Provider` is used.

TestingService_onValidation

Use for testing only.

NevisFIDODatabase_schemaUser

The user which will be used to connect to the database and create the schema (tables).

The database must have been created already (`CREATE DATABASE`) and the user must have `CREATE` privileges for this database.

Example: `schema-user`

NevisAuthRealmBase_signerTrustStore

Defines the trust store nevisProxy uses for validating the signature of the NEVIS SecToken issued by nevisAuth.

If no pattern is assigned automatic key management is asked to provide the trust store. This requires that the `nevisAuth Instance` is part of this project and also uses automatic key management.

Automatic key management should be used for test setups only.

CustomAuthLogFile_syslogHost

Defines where to send logs to via syslog.

This configuration is used only when syslog forwarding is enabled (see `Log Targets`).

The syslog facility is `localhost3` and the threshold is `INFO`.

4.19.0

Full changelog:

Patterns 4.19.0 Release Notes - 2023-05-17

SAML Signature Validation

The SAML IDP now signs the entire SAML `Response` to protect against *XML Signature Wrapping* (XSW) attacks.

This is a breaking change as you have to adapt the configuration of your SAML service providers (SPs) to validate the signature of the `Response` instead of, or in addition to, the `Assertion`.

If this is not possible, you can opt out of this change by selecting `Assertion` in the `Signed Element` drop-down of the `SAML SP Connector`.

If only the `Assertion` is signed, then your setup may be vulnerable to attacks. In this case we recommend to check if your SP applies appropriate mitigations.

If you are using a Nevis SP, then we recommend to upgrade to the latest applicable version of nevisAuth to benefit from additional checks of the `ServiceProviderState`. Check the release notes of nevisAuth for details.

To easily configure which signatures are validated on the SP side, we have added a drop-down `Signature Validation` to the `SAML IDP Connector` pattern.

The default of this drop-down is `both`, which means that the signature of the `Response` and `Assertion` is checked. This in line with the change of the default on the IDP side.

If you can not enable response signing on the IDP site, you can opt out of this change by setting the drop-down to `Assertion`.

OAuth 2.0 Authorization Server / OpenID Provider REST Endpoints

The `REST Endpoints` tab in the `OAuth 2.0 Authorization Server / OpenID Provider` pattern and corresponding settings have been replaced with separate patterns.

You have to adapt your configuration. Add the patterns you need to your project and assign them via the new `REST Endpoints` setting.

nevisProxy upgrade to OpenSSL 3.0

OpenSSL version 3.0 has a more strict default for security level than OpenSSL version 1.1.1. The default security level 1 now forbids signatures using SHA1 and MD5.

In consequence, the following issues may occur:

1. Connections using TLSv1.1 will fail with the following message in the `navajo.log`:

   ```
   3-ERROR : OpenSSL-failure: 00777CC0137F0000:error:0A00014D:SSL routines:tls_process_key_exchange:legacy sigalg disallowed
   ```

   We recommend upgrading your configuration to use TLSv1.2 or TLSv1.3. If it is not possible, you can add the suffix `:@SECLEVEL=0` to your TLSv1.1 cipher suites to allow their signature algorithms.

2. Connections using a certificate with a deprecated signature algorithm will fail with the following message in the `navajo.log`:

```
3-ERROR :  [...] error:0A00018E:SSL routines::ca md too weak (must be pem encoded)) [NVCT-0054]
```

We recommend renewing your certificates with a stronger signature algorithm. In the meanwhile, you can add the suffix `:@SECLEVEL=0` to the cipher suites of the affected filter or servlet. If the issue occurs at several places, or if it affects your EsAuth4ConnectorServlets, you can also modify the default cipher suites to include this suffix. Proceed as follows:

- Add a `Generic nevisProxy Instance Settings` pattern to you configuration.
- Add a `bc.property` for each cipher suite you want to modify. The keys are:
  - `ch.nevis.isiweb4.servlet.connector.http.SSLCipherSuites` for the HttpsConnectorServlets
  - `ch.nevis.isiweb4.servlet.connector.websocket.SSLCipherSuites` for the WebSocketServlets
  - `ch.nevis.isiweb4.servlet.connector.soap.esauth4.Transport.SSLCipherSuites` for the EsAuth4ConnectorServlets
  - `ch.nevis.nevisproxy.servlet.connector.http.BackendConnectorServlet.Secure.CipherSuites` for the BackendConnectorServlets
  - `ch.nevis.isiweb4.filter.icap.ICAPFilter.SSLCipherSuites` for the ICAPFilters
- The modified default values should be `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:@SECLEVEL=0`
- Attach this pattern to your `nevisProxy Instance`, under `Advanced Settings > Additional Settings`.

CustomNevisIDMLogFile_auditLog

Configure audit logging capability of nevisIDM.

- When `JSON` (default) is selected, nevisIDM will write audit entries in JSON format.
- When `plain` is selected, nevisIDM will write audit entries as plain log lines. This setting is deprecated and may be removed in a future release.
- When `disabled` is selected, nevisIDM will not log audit entries at all.

In classic VM deployments the log target is `/var/opt/nevisidm/<instance>/logs/audit.log`. In Kubernetes and when `JSON` is selected the log messages are written to the pod log with the prefix `[audit.log]`.

If you deploy nevisIDM to multiple hosts (multi-instance setup), the audit logging will only be enabled on the first host.

NevisIDMUserCreate_clientId

Enter the client ID where the user shall be created.

SamlIdpConnector_audience

Enter a regular expression to validate the `Audience` of a received SAML `Assertion`.

NevisIDMDatabase_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.

NevisIDMWebApplicationAccess_selfAdmin

Choose between:

- `enabled` - the nevisIDM self admin GUI will be exposed on the path `/nevisidm/selfadmin/`.
- `disabled` - access to the path `/nevisidm/selfadmin/` will be blocked.

If you want to provide a self admin interface for end users we recommend to implement your own application and call the nevisIDM REST API instead. This way you can decide which settings to expose to your users and achieve the desired user experience.

Maintenance_statusCode

The status code of the response with the maintenance page.

By default, the response is sent with status code `503` for `Service Unavailable`.

NevisAdaptDeployable_analyzerConfig

Allows you to customize nevisAdapt Analyzer configuration.

FIDO2Authentication_onCancel

If assigned a button with label 'fido2.cancel.button.label' will be added.

Use to provide an alternative to the user when the user decides to cancel the authentication or the authentication fails and the error cannot be handled.

NevisAdaptAnalyzerConfig_deviceAnalyzer

Device Analyzer is a global setting, disabling this means that the device analyzer will not be used to calculate risk scores. This will result in a lower risk score for all users.

If you wish to disable, consider disabling all other submodules as well.

OAuth2AuthorizationServer_accessTokenFormat

Choose between:

- `JWE` : the access token will be encrypted. This is the default. The token is considered opaque and thus resource servers need to call the token introspection endpoint to validate the token.
- `JWS` : the access token will not be encrypted. Choose this mode to get a signed token which can be validated without calling the token introspection endpoint. A JWKS endpoint will be added to this pattern in 2022.

NevisIDMProperty_accessCreate

Possible settings:

- `READ_WRITE` : Input is possible for the if no previous value was stored.
- `READ_ONLY` : Field is read only.
- `OFF` : Field is not updatable and property is not displayed GUI.

Users with `AccessControl.PropertyAttributeAccessOverride` can edit these field regardless of this settings.

NevisFIDODeployable_clientKeyStore

Assign a key store to be used for the 2-way TLS connection to nevisIDM.

If no pattern is assigned an automatic key store will be generated. This requires automatic key management to be enabled in the inventory. Further, the pattern assigned to `nevisIDM` must be a `nevisIDM Instance` which uses an automatic trust store for the `Frontend Trust Store` .

Note that it is required that the certificate used by nevisFIDO to connect to nevisIDM is uploaded as a certificate credential for the `nevisfido` technical user. This is done automatically when deploying to Kubernetes and using automatic key management on both sides. In any other case, this step has to be done manually.

ErrorHandler_errorPages

Upload HTML error pages, JSON error pages and associated resources here.

Pages must be named like the error code they are used for (e.g. `500.html`). You can use the same page for multiple status code (e.g. `401,403,500-599.html`).

By default, the error pages are deployed to `/errorpages/<name>` but you can set a different location via the property `Base Path` (see `Advanced Settings`).

In your error pages we recommend to use relative links to include resources. You may also include resources deployed on the virtual host via `Hosted Resources`.

The following placeholders are supported:

- `TRANSFER_ID` for the unique ID of the request (e.g. `c0a80e52-5d04-11ac0500-16906714eee-00000003`)
- `TIMESTAMP` to show a timestamp (e.g. `Tue, 19 Feb 2019 15:48:02 GMT`)

NevisAuthDeployable_linePreference

This setting (together with the inventory) defines the order of nevisAuth endpoints in the connection string from nevisProxy.

nevisAuth stores unauthenticated sessions in memory. In a classic deployment to VMs, even when a `nevisAuth MariaDB Remote Session Store` is configured, sessions are synced to the DB only after successful authentication. Thus, multi-step login flows require that requests for the same session are routed to the same nevisAuth endpoint.

nevisProxy uses a simple fail-over strategy. The first URL in the connection string for nevisAuth is always used, unless this instance is not available. This strategy works well when:

- there is only 1 nevisProxy instance
- there are 2 lines of nevisProxy but line 1 is active and line 2 is standby
- there is a session-sticky load-balancer in front of nevisProxy is session-sticky

The order of the connection string depends on the inventory. See also: Defining Lines and Fail-over Association

This strategy may fail in active / active setups when line groups are defined in the inventory. In such setups you can set this drop-down to `disabled` to ensure that the order in the connection string is the same on all nevisProxy lines.

NevisDetectEntrypointDeployable_subPaths

Set to apply this pattern on some sub-paths only.

Sub-paths must be relative (e.g. not starting with `/`) and will be appended to the frontend path(s) of the virtual host (`/`) or applications this pattern is assigned to.

Sub-paths ending with `/` are treated as a prefix, otherwise an exact filter-mapping will be created.

The following table provides examples to illustrate the behaviour:

Frontend Path	Sub-Path	Effective Filter Mapping
`/`	`secure/`	`/secure/*`
`/`	`accounts`	`/accounts`
`/`	`api/secure/`	`/api/secure/*`
`/`	`api/accounts`	`/api/accounts`

Frontend Path	Sub-Path	Effective Filter Mapping
`/app/`	`secure/`	`/app/secure/*`
`/app/`	`accounts`	`/app/accounts`
`/app/`	`api/secure/`	`/app/api/secure/*`
`/app/`	`api/accounts`	`/app/api/accounts`

NevisIDMDeployable_authSignerTrustStore

Assign a Trust Store provider pattern to use for setting up trust between nevisIDM and nevisAuth. If no pattern is assigned the signer key will be provided by the nevisAdmin 4 PKI.

GenericSocialLogin_emailClaim

The claim that contains the e-mail of the logged-in user in the social account. The default value is `email`.

AppleLogin_buttonLabel

Enter the text that should be displayed for the end-user on the social login button, and provide translations for this label on the Authentication Realms.

HostContext_addresses

Define addresses (HTTPS or HTTP) at which this host will be reachable from a client perspective.

The basic syntax is:

- `<scheme>://<hostname>`
- `<scheme>://<hostname>:<port>`

A variable may be used to define different addresses for different stages (e.g. DEV, TEST, PROD).

The expression `${deployment_host}` may be used when the name of the target host is required.

Examples:

- `http://www.siven.ch`
- `https://www.siven.ch`
- `http://${deployment_host}:8080`

The `port` will, if omitted, default to `443` for HTTPS and to `80` for HTTP.

You also have to set `Bind Addresses` if:

- the addresses cannot be resolved on the target host(s)
- the port should be opened on different addresses / IPs, or ports.
- multiple virtual hosts should listen on the same endpoint (name-based virtual hosts).

NevisIDMGenericBatchJob_resources

Upload JAR file(s) for custom batch jobs.

Note that batch jobs which call the nevisIDM business layer are not supported by Nevis. Please call the nevisIDM REST API only.

NevisAdaptEvent_events

Select at least one event for the combination to react on:

- `unknown-device` : this is the first time for this device cookie
- `unknown-country` : this is the first time for this geolocation (country)
- `unknown-fingerprint` : this is the first time for this browser fingerprint
- `suspicious-country` : the login request came from a prohibited country
- `high-ip-velocity` : the current geolocation is physically too far to be reachable since the last login
- `ip-reputation-blacklisted` : the login request came from an IP address with low reputation

For technical details check Event-based configuration.

AuthenticationConnectorStepBase_onSuccess

Set the step to continue with on successful authentication.

AccessRestriction_override

By default, access restriction rules apply to all sub-locations.

For instance, when you assign an `Access Restriction` pattern to a `Virtual Host` all applications on this virtual host will be affected.

To **replace** the rules defined on a parent location select `enabled` on all `Access Restriction` patterns in the hierarchy.

If `disabled` is selected anywhere in the hierarchy the rules are considered **additional**.

Technical Details:

This feature is implemented using a nevisProxy `LuaFilter` . Mapped filters are inherited to sub-locations unless an `exclude-url-regex` is defined.

By selecting `enabled` the generator is informed that the mapped filter has the purpose `access restriction`. The generator then ensures that an `exclude-url-regex` entry is generated when a filter with the same purpose is mapped to a sub-location.

NevisIDMProperty_description

The `description` field in the property definition file allows you to provide a clear and informative description of the custom property. This description will be valuable for understanding the purpose, expected values, or any other relevant information about the property.

The description will be escaped for JSON if required.

GenericNevisProxySettings_parameters

Define *Template Parameters*.

Examples:

```
backend-host: backend.siven.ch
```

These parameters can be used in:

- `Configuration: navajo.xml`
- `Configuration: bc.properties`

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 `\}` .

SamlResponseConsumer_logoutProcess

Assign a step to apply custom post-processing logic which is executed when a `LogoutRequest` or `LogoutResponse` message is received.

OAuth2AuthorizationServer_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 `Custom Transitions` .

GenericSocialLogin_providerEndpoint

The provider endpoint that contains the configuration of the OpenID Connect server. It's required when `providerType` has the value `OpenID Connect` .

SamlSpRealm_tokens

SAML Responses returned by the IDP are consumed in nevisAuth and **not** forwarded to applications.

If your application requires a token then you have assign a pattern which can produce that token here. For instance, assign a `NEVIS SecToken` or `SAML Token`.

To forward the token to applications you also have to assign the token pattern to these applications via `Application Access Token`.

The token will be created on first access (missing token role triggers a stepup). In case of a session upgrade via SAML the token (role) is revoked and thus the token is recreated on the next access.

In your application you may use the Ninja authentication filter provided by NEVIS to extract user id, roles, and custom attributes.

SamlSpConnector_properties

Configure properties of the nevisAuth `IdentityProviderState`.

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

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

Examples:

Key	Value
out.extension.Bearer	ch.nevis.esauth.auth.states.saml.extensions.SubjectConfirmationExtender
Bearer.inResponseTo	${notes:saml.request.id}
out.signatureKeyInfo	Certificate

NevisAuthDeployable_host

Enter a custom host name to listen on.

This setting is relevant in classic VM deployment, when working with multi-homed target hosts.

In Kubernetes nevisAuth listens on `0.0.0.0` and thus this setting is discouraged.

HeaderCustomization_basicAuthPass

Enter the basic auth password or an expression of the format `<source>:<parameter>` .

For the `<source>` you may use:

- `AUTH` : outargs returned by nevisAuth.
- `CONST` : constant strings.
- `ENV` : Apache environment variables.
- `PARAM` : values from a request body as provided by a `ParameterFilter` .
- `HEADER` : request headers.

LuaPattern_parameters

Parameters defined here can be used inside the Lua script.

The name of each parameter must start with `param_` . This limitation may be lifted in a future release.

The value will be trimmed.

Set this property if you need a **different** value depending on the inventory.

1. click `var` to use a nevisAdmin 4 variable for the **entire** setting:
    - Enter a good name for the variable as the default may be quite verbose.
    - Enter some sample values to document the variable in the project.

2. add the nevisAdmin 4 variable to your inventories:
    - See below for an example which illustrates the syntax.

Example inventory variable:

```
vars:
  example-variable:
    param_example_string: "on"
    param_example_numeric: 60
```

It is sometimes required to quote values. In the example above, the value `on` would be converted to a boolean value if it weren't for the double quotes `"` . When unsure, always put double quotes around the value.

OAuth2AuthorizationServer_accessTokenClaims

Configure additional claims for the OAuth2.0 Access Token.

Claims are added if they have a value.

For instance, claims may be added when a certain scope is requested which includes them.

OpenID Connect defines the following `scope` values which may be requested to get `claims` :

- `profile` . claims: `name` , `family_name` , `given_name` , `middle_name` , `nickname` , `preferred_username` , `profile` , `picture` , `website` , `gender` , `birthdate` , `zoneinfo` , `locale` , `updated_at` .
- `email` . claims: `email` , `email_verified`
- `address` . claims: `address`
- `phone` . claims: `phone_number` , `phone_number_verified`

Examples:

```
given_name=${sess:ch.nevis.idm.User.firstName}
family_name=${sess:ch.nevis.idm.User.name}
email=${sess:ch.nevis.idm.User.email}
mobile=${sess:ch.nevis.idm.User.mobile}
```

NevisIDMPasswordLogin_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 please 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>}`.

NevisIDMUserCreate_onFailure

Define how to continue after user creation, if it was unsuccessful.

ResponseRewritingSettings_responseRewrite

- `off` disables automatic response rewriting
- `header` enables auto rewrite of response headers (includes cookies)
- `complete` enables auto rewrite for response headers and body

GenericSocialLogin_clientExtId

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

NevisDetectRiskPluginBase_trustStore

Reference a trust store provider pattern or leave empty to manage the trust store with nevisAdmin.

RealmBase_cookieSameSiteRelaxation

Some older browsers treat cookies with `SameSite=None` as Strict.

See this example bug report for Safari:

Bug 198181 - Cookies with SameSite=None or SameSite=invalid treated as Strict

Enable this feature to map a filter to the root location `/*` which evaluates the `User-Agent` request header to remove `SameSite=None` for browsers which are known to be affected.

LogSettingsBase_rotationInterval

Rotation interval after which log files are rolled over.

This configuration is *not* used when `Rotation Type` is set to `size`.

Choose between:

- `daily` - the postfix of rotated files will be `.%d{yyyy-MM-dd}`
- `hourly` - the postfix of rotated files will be `.%d{yyyy-MM-dd-HH}`

NevisProxyDatabase_mode

Select one of:

- `classic` - sessions are stored in the remote database only. Recommended setting for production setups.
- `hybrid` - adds a local cache to improve the performance of the session store. This value is experimental and should **only** be used for test setups. This mode requires a session-sticky load balancer in front of nevisProxy. The generated configuration may change in future versions.

NevisProxyDatabase_peer

The hybrid session store requires that the `nevisProxy Instance` is deployed on 2 lines. For illustration purposes let's call the hosts where the instances are deployed `p1` and `p2`.

Enter the URL where the other nevisProxy `Virtual Host` exposes its local session store. The URL must be reachable and should not go via a load-balancer to ensure that the request reaches the peer proxy directly.

You can use variables to ensure that the correct host name is used for the configuration on each line. For instance, the variable may be a host variable and have the following values:

- for server `p1` use: `https://p2:443`
- for server `p2` use: `https://p1:443`

Alternatively, you can use a semantic host name and and define this name in `/etc/hosts` on both `p1` and `p2`.

Example: `https://proxy-peer:443`

OutOfBandMobileRegistration_username

The `username` is used by nevisFIDO to look up the user in nevisIDM.

Depending on how the `nevisFIDO FIDO UAF Instance` is configured, either the `extId` or the `loginId` have to be used.

NevisIDMAdvancedSettings_javaOpts

Add additional entries to the JAVA_OPTS environment variable.

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

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

```
-XX:+HeapDumpOnOutOfMemoryError
-XX:HeapDumpPath=/var/opt/nevisidm/${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.

NevisAuthDeployable_connectionHost

Enter the host:port pair(s) which nevisProxy should use to connect to this nevisAuth instance.

This setting is required when the primary names of the nevisAuth hosts are not accessible by nevisProxy, which is sometimes the case in classic VM deployment, when working with multi-homed target hosts.

The server certificate provided by the `Frontend Key Store` must be valid for all provided hosts names.

OAuth2Scope_description

Used by the ID Cloud management console to store a description for this scope provided by the user.

NevisIDMPasswordLogin_entryPath

The path prefix of the links for the password forgotten process.

Example: given a domain `www.adnovum.ch` and the value `/pwreset/`, all password forgotten steps will use the base path `www.adnovum.ch/pwreset/`.

NevisAuthRealm_langCookieDomain

Enter a domain for the cookie that nevisLogrend issues to remember the language of the user.

This setting should only be used when you want to issue a *wildcard cookie* to share the language with other sub-domains (e.g. across multiple `Virtual Host`).

For instance, if you enter `.example.com` then the cookie will also be sent to `subdomain.example.com`.

NevisAuthRealmBase_sessionValidation

A newline separated list of rules declaring attributes that must not change in the same session. A rule has the following syntax:

```
AUTH|ENV|CONST|PARAM|HEADER:<name of the attribute>:block|invalidate
```

- `block`: the request will be blocked and `403 (Forbidden)` will be returned
- `invalidate`: the session will be invalidated and a new one will be created

nevisProxy Conditions are supported. See nevisProxy reference guide for details.

For instance, use the following configuration to terminate the session if the source IP changes:

```
ENV:REMOTE_ADDR:invalidate
```

SamlToken_subject

Configure the subject of the generated SAML assertion.

- `User ID`: sets the internal user ID
- `Login ID`: sets the ID as entered by the user during login

DatabaseBase_connectionUrl

Set **only** if you have to use a JDBC connection string which the pattern cannot generate.

If the prefix of the connection string works for you and you **only** have to add or overwrite query parameters, set `Connection Parameters` instead.

If you have to use this setting, please consult your setup with your integration partner.

In Kubernetes deployments the connection string configured here is used by the component **only**. It is **not** used to set up and migrate the database schema.

Thus, this setting should **only** be used in classic deployments, or when `Database Management` is `disabled`.

CustomNevisIDMLogFile_auditLogFormat

Log4j 2 log format for the default SERVER logs.

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

NevisIDMPasswordLogin_legacyLitDictMode

In legacy mode policy violations are displayed using 1 GUI element.

You can use `enabled` here until November 2021 when this mode will be removed.

Logout_redirect

Enter a URL or path to redirect to after logout.

CustomNevisIDMLogFile_applicationSyslogFormat

Log4j 2 log format for the SERVER SYS logs.

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

SamlIdp_errorRedirect

URL to redirect to when the IDP is unable to handle the request.

There are 3 cases:

- a session is required for the current operation (e.g. logout, session upgrade) but no session was found.
- the authentication type (SP-initiated or IDP-initiated) is not allowed.
- not enough information for IDP-initiated authentication (e.g. missing query parameters).

If no URL is configured, the IDP will redirect back to the `Referer`, or `/` if no `Referer` header has been sent.

EmailTAN_sender

Sender email address.

TCPSettings_keepAliveConnectionPoolSize

Number of pooled TCP connections. A TCP connection is only put in the pool if the size of the pool does not exceed the configured size. By leaving this field empty, you will be using the nevisProxy default value.

PropertiesTestPattern_dateTimePickerProperty

Not sure why we don't have a date time picker yet.

NevisLogrendLogSettings_levels

Configure log levels.

See nevisLogrend Technical Documentation, chapter Logging configuration 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.logrend.beans.LoginBean = DEBUG
```

NevisAuthDatabase_encryption

Enables SSL/TLS in a specific mode. The following modes are supported:

- `disabled` : Do not use SSL/TLS (default)
- `trust` : Use SSL/TLS for encrypted transfer. Do not perform certificate or hostname verification. This mode is not safe for production applications but still safer than `disabled` .

- `verify-ca` : Use SSL/TLS for encryption and perform certificate verification, but do not perform hostname verification.
- `verify-full` : Use SSL/TLS for encryption, certificate verification, and hostname verification.

SamlSpConnector_preProcess

Assign a step to apply custom pre-processing logic before validating the incoming request for this SP.

You may assign a chain of steps to build a flow.

The flow will be executed for all incoming requests, no matter if the user has a session already.

If you need to apply different logic for these 3 cases you can use `Dispatcher Step` and dispatch based on the following expressions:

```
${request:method:^authenticate$:true}
${request:method:^stepup$:true}
${request:method:^logout$:true}
```

The dispatching will continue after 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 as success exits by assigning the `Pre-Processing Done` pattern.

SamlSpRealm_spLogoutMode

Defines how this SP should react when an SP-initiated logout completes on this SP.

- `redirect-target` : redirects to a defined path or URL. When this option is selected a `Logout Target` must be entered.

- `redirect-state` : redirects according to the `RelayState` query parameter received in combination with the `LogoutResponse` . The IDP is expected to return this parameter as-is and thus the `RelayState` should contain the URL where the logout was initiated. As this is a protected application URL authentication will be enforced and the user will be sent to the IDP again to perform a login.

KerberosLogin_keyTabFile

Upload the Kerberos keytab file.

nevisAuth uses this file to validate Kerberos tokens sent by browsers.

Please check the nevisAuth Technical Documentation on how to create the keytab file.

The keytab file will be deployed to the `conf` directory of the nevisAuth instance.

For a more secure and environment-specific configuration you have the following alternatives:

- create a variable and upload the keytab file in the inventory
- set `Keytab File Path` instead of uploading the file and deploy the file by other means

In complex setups with multiple `Kerberos Realms` and/or `Frontend Addresses` you may have to upload multiple keytab files.

SamlIdpConnector_binding

Configure the outgoing binding. This affects how the SAML `AuthnRequest` is sent to the IDP.

NevisDPDeployable_customJars

Upload custom JAR files to handle specialized logic.

SamlSpRealm_logoutReminderPage

Enable this feature to show a logout reminder page.

The page will be shown on next access in the following cases:

- the user has closed the browser
- user session has expired due to idle timeout

The page contains a heading, an info message and a continue button. You can customize them via `Custom Translations` by setting the following labels:

- `title.logout.reminder`
- `info.logout.reminder`
- `continue.button.label`

For this feature to work an additional cookie `Marker_<name>` will be issued. The value will be set to `login` or `logout` depending on the last action of the user.

The following requirements must be fulfilled:

- Usage of HTTPs to access the application and for the entire SAML process.
- No other session expiration feature must be used.

OAuth2Client_responseTypes

Enter the allowed response types.

InitRuntimeConfiguration_configurations

Add key/value pairs to initialize runtime configurations in a session variable.

- Key: name of a nevisAuth session variable
- Value: name of a nevisAdmin4 inventory/project variable.

The inventory/project variable is replaced by its groovy-escaped String value and will be set to the session in a generated groovy script.

```
- ch.nevis.idc.config.authentication.accessAppEnabled: "accessAppEnabled"
- ch.nevis.idc.config.authentication.passkeyEnabled: "passkeyEnabled"
```

NevisFIDODeployable_firebaseProxyAddress

The URL of the HTTP/HTTPS proxy used by nevisFIDO to access the Firebase Cloud Messaging service.

Note: The FCM dispatcher requires outbound access to the Google API service, specifically https://oauth2.googleapis.com for authentication and https://fcm.googleapis.com for accessing the FCM HTTP API. In case proxies and/or company firewalls are in place the connectivity to these Google services must be ensured.

TLSSettings_protocols

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

Check the Apache Documentation for details.

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

```
-all +TLSv1.2 -TLSv1.3
```

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

NevisIDMPasswordLogin_resetLockedPassword

Defines whether it is possible to reset locked passwords or not.

- If enabled, it is possible to reset locked passwords as well. In this case, only disabled passwords cannot be reset.
- If disabled, it is only possible to reset active passwords.

GenericDeployment_deploymentHosts

The host group to deploy the `Files` to and execute the `Command` on. For testing purposes you can also enter a host **name** instead of a group.

The host name / group must exist in the selected inventory.

KeyObject_type

Select `key store` when a private key is needed. Select `trust store` for providing trusted certificate (e.g. for signature validation).

GenericSocialLogin_scope

The request scope(s) for getting the user information from the social account. The default value is `email`.

The scope `openid` will be added automatically if `providerType` is set to `OpenID Connect`.

Scope `offline_access` for generate refresh token.

NevisProxyObservabilitySettings_deploymentEnv

Allows the configuration of the `deployment.environment` key-value pair resource attribute.

NevisIDMDeployable_logging

Add logging configuration for nevisIDM.

README

nevisAdmin 4 Plugins

Plugins to use and configure NEVIS from nevisAdmin 4.

Setup

Use Java 17 JDK (openjdk).

To resolve dependencies from private repositories (nevisAdmin 4 BE) you have to create a personal access token for your github account.

Don't forget to "Enable SSO" for this token!

Now you can configure the user and token once and for all in your global gradle.properties.

```
cat ~/.gradle/gradle.properties
GITHUB_USER=benjamin-koenig
GITHUB_TOKEN=...
```

Commit signing is enforced in this repository for all branches. Configure your GIT client according to our best practises.

You can validate the signature of a commit using:

```
git verify-commit <commit>
```

Build

- Build all sub projects

  ```
  ./gradlew assemble
  ```

- Build using a certain version

  ```
  GITHUB_RUN_NUMBER=1 ./gradlew assemble -Dorg.gradle.project.BUILD_VERSION_BASE=4.9.1  GITHUB_REF=refs/tags/release/4.9.1.2
  ./gradlew assemble
  ```

- Clean and builds projects

  ```
  ./gradlew clean build
  ```

- Run the unit and architecture tests

  ```
  ./gradlew unitTest
  ```

- Run integration tests (nevisadmin-test-system)

```
./gradlew systemTest
```

These tests rely on the `nevis-dev-systemd` docker image. If you have problems with this image check with the team.

- Start nevisAdmin 4:

  in the `nevisadmin4` repository run `./gradlew devRun` and keep the task running

- Deploy to local nevisAdmin 4:

  ```
  ./gradlew deploy
  ```

- Copy plugins into a folder for easy upload

  ```
  find . -name nevisadmin-plugin*4.9.1.1.jar -exec cp {} /tmp/ \;
  ```

IDEA

- File / Open / Select the build.gradle
- In general delegate all build actions to the gradle wrapper.

Maintenance

Upgrade Gradle Wrapper

```
./gradlew wrapper --gradle-version=6.2 --distribution-type=bin
```

Backport

`git cherry-pick` seems to be the most robust way to backport changes to the `release/*` branches.

Github Packages Housekeeping

There is a cleanup workflow: `.github/workflows/cleanup.yml`

You can also use the tools provided by the `housekeeping` repository to delete old versions which are not required anymore.

```
./gradlew delete_packages -Dgithub_repo=nevisadmin4-plugins -Dmaven_version=<version>
```

Have a look at the plugin-base to find out which versions exist:

https://github.com/nevissecurity/nevisadmin4-plugins/packages/142340/versions

You can remove all versions which:

- are not referenced by any release branch of `nevisadmin4` repository (`release/4.5` and `master`)
- have not been handed out to customers for testing purposes

Release Process Details

Create a new pre-release in Github: https://github.com/nevissecurity/nevisadmin4-plugins/releases/new

For the tag version enter: `release/<version>`

Check `This is a pre-release` .

The `.github/workflow/release.yml` will now be executed. When the workflow completes the JARs are published in Github packages.

Download Github Packages

You can use the following steps to download JARs of a certain released version from Github packages into `target/download` :

```
rm target/download/*
```

Now use either:

```
GITHUB_REF=refs/tags/release/4.9.1.2 ./gradlew download
```

or:

```
GITHUB_RUN_NUMBER=1 ./gradlew download -Dorg.gradle.project.BUILD_VERSION_BASE=4.9.1
```

Cloudsmith Publish Tasks

Cloudsmith publishing is done as part of the release process when the checkbox `This is a pre-release` is **not** selected.

You can also do these steps manually.

For instance, you can use the `helper/cloudsmith-push.sh` script to push all JARs to the `delivery` repository. This is the highest repository at Nevis which is **not** customer visible.

In order for customers to download these JARs they have to be tagged with a quarterly label (e.g. `2023R2`). The tag is defined in `gradle.properties` and must be incremented after each quarterly releases on the `main` branch.

Use the following task to apply the current tag to artefacts in delivery:

```
CLOUDSMITH_TOKEN=... GITHUB_REF=refs/tags/release/4.20.0.1 ./gradlew tagDelivery
```

Then the JARs in Cloudsmith `delivery` then need to be copied to `rolling` . Because of costs it is better to move them instead.

There is no Gradle task for that as this copy is usually done by a workflow in `neviscluster` , or manually for intermediate releases.

Check Documentation Links

The help of a pattern often contains links to docs.nevis.net which break easily. New URLs won't be available on the productive docs.nevis.net as they become available on release day when the integration branch is merged into main and docs is redeployed. Hence, we have to check manually (for now) whether the links still point to an existing page.

There is a Python script that can be simply run in a terminal as `helper/check-docs-urls.py` and it will generate warnings about which URLs in which files are broken.

RequestValidationSettings_customRules

Configure *exception modifications*.

As explained in the ModSecurity documentation *exception modifications* are applied **after** including the core rules.

Note that new rule may require a rule ID which has to be unique for this pattern. Use the range 1-99,999 as it is reserved for local (internal) use.

- Remove rule with ID `900200` :

```
SecRuleRemoveById 900200
```

- Whitelist body parameter `upload` for all rules:

```
SecRuleUpdateTargetByTag ".*" "!ARGS:upload"
```

- Whitelist body parameter `upload` for rule ID `123` :

```
SecRuleUpdateTargetById 123 !ARGS:upload
```

- Add a new rule which allows the HTTP methods used for WebDAV:

```
SecAction \
 "id:1,\
  phase:1,\
```

```
    nolog,\
    pass,\
    t:none,\
    setvar:'tx.allowed_methods=GET HEAD POST OPTIONS PUT PATCH DELETE CHECKOUT COPY DELETE LOCK MERGE MKACTIVITY MKCOL MOVE PROP
```

NevisAdaptPluginPattern_properties

Set the value for the following optional parameters if the default ones do not match the requirements:

- cacheDisabled = (default 'false')
- ignoreHttpRequest = (default 'false')
- ignoreTlsObservation = (default 'true')

GenericIngressSettings_path

Define a custom path for the generated ingress resource.

Example:

```
/nevis/
```

This is an ingress specific setting, the endpoints have to be configured separately to be available under the defined path. When using side-by-side deployment, the path must be the same between the primary and secondary deployment.

OnDemandEntry_onEntry

Point to the first step of the authentication process.

NevisDetectDatabase_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.

NevisFIDO2Database_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.

NevisAdaptDatabase_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.

NevisDPDeployable_logging

Add logging configuration for nevisDataPorter.

NevisAdaptDatabase_oracleVolumeClaimName

Due to licensing restrictions, we cannot ship any Oracle dependencies.

If you are using an Oracle database, are deploying to Kubernetes, and `Database Management` is *enabled* (`complete` or `schema`), then you have to provide a Kubernetes volume containing an Oracle driver and client.

For more information, see Preparing Oracle Volume.

Enter the name of that volume here.

The volume will be mounted in the `nevisadapt-dbschema` image to set up and patch the database schema.

The volume will be mounted in the `nevisadapt` image to connect to the database. Because of that, there is no need to upload a `JDBC Driver` .

LogSettingsBase_syslogHost

Defines where to send logs to via syslog.

This configuration is used only when syslog forwarding is enabled (see `Log Targets`).

The syslog facility is `localhost3` and the threshold is `INFO` .

NevisIDMSecondFactorSelection_fido

Assign a step which may be selected when the user has an FIDO UAF Authenticator credential.

For instance, assign the `Out-of-band Mobile Authentication` pattern.

CustomAuthLogFile_serverLog

Select the type of log4j appender.

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 a Syslog server
- `syslog` - forward to a Syslog server only

KeyObject_keyStore

Reference a key store provider pattern or leave empty to let nevisAdmin establish a key store. This reference property is considered when type `key store` is selected.

EmailTAN_testingMode

When testing mode is enabled the TAN code is always `AAAAA` .

Thus, you can test the flow more easily during integration.

No email will be sent and thus no `SMTP Server` needs to be assigned.

NevisFIDODeployable_backendTrustStore

The trust store nevisFIDO uses to connect to `nevisIDM Instance` .

TANBase_testingMode

Enables "Testing Mode". The TAN challenge is `AAAAA` .

When testing mode is enabled, the TAN challenge is constant and might not be sent over the linked connection.

Each connection pattern decides individually how it behaves with respect to "Testing Mode".

For instance, the `SwissPhone Connection` does not sent a message to the gateway when "Testing Mode" is enabled.

WebhookCalls_login

Configure Webhook calls for login flows.

NevisIDMPasswordLogin_emailSentRedirect

Where to redirect to once the password reset ticket has been generated.

- `root` : to the domain root (`/`) on this `Virtual Host`
- `referrer` : to the initial URL requested by the client

- `custom` : to a custom path or URL as configured by `Custom Email Sent Redirect`

Note that the `referrer` will always be a page requiring authentication, hence it will basically redirect to the login page.

OAuth2Client_tokenEndpointAuthMethod

Set authentication method for Token Endpoint.

- None: for public client without secret
- Client_secret_basic: for confident OAuth 2.0/OpenId Connect Client. Client need to send combination between OAuth 2.0/OpenId Connect Client and Secret in base64 using Authorization Header when call to Token Endpoint.
- Client_secret_post: for confident OAuth 2.0/OpenId Connect Client. Client need to send OAuth 2.0/OpenId Connect Client and Secret in request body when call to Token Endpoint.

NevisIDMAuthorizationsAddon_roleManagementFile

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

See Assigning IDM roles for details.

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

NevisIDMChangePassword_encryption

Set to enable form encryption.

This feature is still experimental in nevisAdmin 4 and has been added to this pattern as a preview.

The default template includes the required JavaScript (e2eenc.js) to perform client-side encryption of the form values.

WebApplicationAccess_responseRewrite

Enable to replace backend host names in responses or set to `custom` for complex rewriting use cases.

- `off` - disables automatic response rewriting
- `header` - enables auto rewrite for response headers (including `Set-Cookie` header)
- `complete` - enables auto rewrite for the entire response (including body)
- `custom` - requires assignment of `Response Rewriting Settings` via `Additional Settings`

NevisFIDOServiceAccess_fido

Assign a `nevisFIDO FIDO2 Instance`.

OATHAuthentication_onSuccess

Configure the step to execute after successful authentication.

If no step is configured here the process ends and the user will be authenticated.

OutOfBandMobileStepBase_host

To complete the authentication, the mobile app will send a request to `/nevisfido/token/redeem/authentication`.

The domain is coded into the mobile app and has to be communicated when ordering the app.

We recommend to assign the `Virtual Host` which serves that domain here so that this pattern can generate the required configuration.

The `Virtual Host` assigned here will also be considered when calculating the `Frontend Address` in the `nevisFIDO UAF Instance`.

NevisDetectDeployableBase_jmsClientTrustStore

Reference a trust store provider pattern or leave empty to manage the trust store with nevisAdmin.

AuthCloudOnboard_onboardingScreenButton

Adds another button to the onboarding screen.

The button may have a special `Button Name` set to render it in a nice way using 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.

NevisAdaptDeployableBase_trustStore

Reference a trust store provider pattern or leave empty to manage the trust store with nevisAdmin.

MultipleFieldUserInput_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.

GroovyScriptStep_guis

Add `Gui` elements to the `Response`.

For each line 1 `Gui` element will be generated.

Most authentication states have only 1 `Gui` element.

The format is key-value pairs. The key is used as `name`. The value is optional and used as `label`.

For instance, the line `auth:title.login` will produce the following `Gui` element:

```
<Gui name="auth" label="title.login"/>
```

Configuration of `GuiElem` elements is not supported. You have to create them dynamically in your script.

Here is an example how to render a certain `Gui` and add `GuiElem` elements:

```
response.setGuiName('login')
response.addInfoGuiField('info', 'info.login', null)
```

RequestValidationSettings_logOnlyMode

Allows to use the request validation settings in log only mode.

AccessRestriction_countryRules

Defines what action should be taken for a specified country.

Possible actions are:

- **allow**: Requests are let through
- **log**: A log entry is made for each request from the specified country

- **block**: Blocks requests from a country

NevisFIDODeployable_relyingPartyId

Enter a base domain for all `Relying Party Origins`.

Example: `example.com`

PropertiesTestPattern_modSecurityRuleProperty

Used in 1 place only.

OATHAuthentication_loginType

Sets the type of login identifier which will be used to look up the user.

In nevisIDM any client whose users should be able to log in with their email address must have the following entry in the Client policy:
`authentication.loginWithEmail.enabled=true`

OutOfBandMobileStepBase_nevisfido

Assign a `nevisFIDO UAF Instance` pattern. nevisFIDO provides required services for out-of-band authentication.

NevisAuthDeployable_dependencies

In case `AuthStates` uses custom AuthState classes upload the required JAR file(s) here. Files will be deployed into the `plugin` directory of the nevisAuth instance.

AzureServiceBus_provisioning

Remote Azure Service Bus Queue to which provisioning messages should be sent.

SocialLoginCreateUser_idGeneration

Define how the user extId and profileExtId are generated. Choose between:

- undefined: the extId and profileExtId will be generated by nevisIDM, and it depends on nevisIDM config that the id will be uuid or in any sequence
- uuid: the extId and profileExtId will be generated in UUID format by nevisAuth and send to nevisIDM. The newly created user extId and profileExtId will be in UUID format

NevisIDMDatabase_oracleApplicationRoleName

Name of the application role for the oracle database. It's recommended to keep the default value unless the pattern is used with an existing database that has a different one.

NevisAdaptDatabase_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.

NevisFIDOLogSettings_levels

Configure log levels.

In classic deployment nevisAdmin 4 does **not** restart nevisFIDO if you only change log levels. The log configuration will be reloaded within 60 seconds after deployment.

The category `ch.nevis.auth.fido.application.Application` will **always** be generated. If you don't set its level, `INFO` will be used.

This gives you:

- log messages during startup and when the startup is done
- 1 line per incoming request
- 1 line for each API call towards nevisIDM

Debug incoming requests:

```
org.springframework.web.filter.CommonsRequestLoggingFilter = DEBUG
```

Debug the entire component:

```
ch.nevis.auth.fido = DEBUG
```

AuthCloudBase_allowedOperations

You can customize the pattern behavior by choosing one of the allowed operations.

The pattern supports two operations:

- enrollment: registers the user and their mobile device in the Authentication Cloud Instance.
- login confirmation: authenticates the user by sending a confirmation request to their mobile device via push notification.

Available options:

- enrollment: Choosing this will *disable login confirmation*.
- login confirmation: Choosing this will *disable enrollment*. Make sure the user and their mobile device are *already registered* in the Authentication Cloud Instance.
- both (default): Allows both operations.

NevisAuthRealmBase_template

Customize the rendering of login pages.

Download the default template to get started.

nevisLogrend: Simple Mode

Point your browser to a protected application to have a look at the login page. Download any resources (e.g. images, CSS) that you want to adapt. Then upload the changed files here.

To change the outer HTML upload a file named `template.html`. Here is a simple example:

```
<!DOCTYPE html>
<html lang="${lang.code}">
  <head>
    <title>${label.title}</title>
    <link href="${resources}/bootstrap.min.css" rel="stylesheet" type="text/css">
    <link href="${resources}/default.css" rel="stylesheet" type="text/css" media="all">
```

```
    </head>
    <body>
      <header id="header" class="container-fluid">
        <img class="logo center-block" src="${resources}/example.svg" alt="NEVIS Security Suite">
      </header>
      <main id="content" class="container">
        ${form}
      </main>
    </body>
  </html>
```

Please also upload file resources referenced by your template (e.g. images, CSS, Javascript). Use this when you reference additional files, or if you want to override the default files provided.

The template must contain `${form}` and may contain additional expressions.

Expression	Description
`${form}`	generated login form (required)
`${lang.switch}`	language switcher component
`${lang.code}`	current language code (i.e. EN, DE)
`${label.title}`	a human-readable title
`${label.myLabel}`	a custom text which must be defined via `Custom Translations`
`${resources}`	path to static resources (e.g. CSS, images, Javascript)

Some resources (i.e. bootstrap.min.css, default.css) are provided out of the box because they are required by the default template. Feel free to use them.

nevisLogrend: Expert Mode

Expert users may upload Velocity templates and resources to nevisLogrend.

Zip files will be extracted into the nevisLogrend *application*:

```
/var/opt/nevislogrend/<instance>/data/applications/<realm>
```

Flat files will be added to the following subdirectories:

- `webdata/template` : Velocity templates (`*.vm`)
- `webdata/resources` : additional resources (e.g. images, CSS)

nevisProxy: Simple Template

nevisProxy provides a simple login page renderer which can be used instead of nevisLogrend. See `Login Renderer` for details.

For each enabled language (e.g. `en`) upload a file named `<lang>_template.html` . The template must contain the placeholder `NEVIS_AUTH_FORM` .

If your templates require additional resources (e.g. CSS, images) upload them as `Hosted Resources` on the nevisProxy virtual host.

NevisIDMSecondFactorSelection_mTAN

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

You can assign any step here but we recommend to use the `Mobile TAN` pattern.

The session variable `user.mobile` will contain the mobile number from the mTAN credential.

BehavioSecPluginPattern_fraudulentFlags

List of BehavioSec report flag names. Please add each entry line-by-line.

If any of these flags contains true value in the report, the request is marked as fraudulent and the request fails.

If the field remains empty, the items marked with (*) will be part of the default configuration.

Potential flag names (as of 5.4):

- advancedUser (*)
- autoModel
- coached (*)
- deviceChanged (*)
- deviceIdShared (*)
- deviceIntegrity (*)
- diError (*)
- drFlag (*)
- finalized
- ipChanged (*)
- ipShared (*)
- isDataCorrupted (*)
- isBot (*)
- isDuplicate (*)
- isOneHand
- isRemoteAccess (*)
- isReplay (*)
- isSessionCorrupted (*)
- isWhitelisted
- locationMismatch (*)
- newCountry (*)

- newsubprofile
- numpadAnomaly (*)
- numpadUsed
- numrowUsed
- ohFlag (*)
- otjsError (*)
- pdError (*)
- pnFlag (*)
- pocAnomaly (*)
- pocUsed
- tabAnomaly (*)
- tabUsed
- travelTooFast (*)
- uiConfidenceFlag (*)
- uiScoreFlag (*)

WebhookCalls_signup

Configure Webhook calls for signup flows.

Logout_label

Enter a label for the message that shall be presented to the user. This is used when `Logout Behaviour` is set to `gui`.

PropertiesTestPattern_durationProperty

Enter a time duration with unit.

DatabaseBase_keyStore

Define the key store to use for 2-way HTTPs connections for DB endpoint.

This configuration only accept PEM Key Store pattern configuration.

Noted: This is an experimental configuration

NevisAdaptAnalyzerConfig_fingerprintAnalyzer

Fingerprint Analyzer is a global setting, disabling this means that the device analyzer will not be used to calculate risk scores. This will result in a lower risk score for all users.

If you wish to disable, consider disabling all other submodules as well.

NevisDetectLogSettings_serverLogFormat

Logback 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`.

AuthCloudBase_onSkip

Assign a step to continue with when the user clicks the skip button.

A skip button will be added to the authentication screen.

AuthenticationFlow_stepup

Assign a step to execute for incoming requests when there already is an authenticated session.

If not present already, the step will be added to the `Authentication Realm`.

If no step is assigned the same step as for `Authentication Flow` will be executed.

CookieCustomization_sessionCookies

Cookies listed here will be stored in nevisProxy.

However, cookies marked as `Client Cookies` in any `Cookie Customization` pattern assigned to the same application will still be allowed to pass through!

Storing cookies requires a user session. Thus, we recommend to not use this feature for stateless or public applications!

Incoming cookies with the same name will be blocked.

Regular expressions are supported.

Example:

- `.*SESSION.*`

NevisFIDO2Database_schemaUser

The user which will be used to connect to the database and create the schema (tables).

The database must have been created already (`CREATE DATABASE`) and the user must have `CREATE` privileges for this database.

Example: `schema-user`

NevisProxyDatabase_type

Choose between `MariaDB` and `PostgresSQL` .

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

OAuth2AuthorizationServer_clients

Assign `OAuth 2.0 Client` patterns.

Configuration is ignored if `nevisMeta` is used.

PropertiesTestPattern_portProperty

Enter a port number.

SwissPhoneChannel_password

The password to use to connect to the SwissPhone SMS Gateway.

TANBase_maxRegenerate

The maximum number of times a **new** code can be generated.

If the value is `1` or greater, a *resend* button will be added to the screen. The button is shown only when there are still resends left.

When you configure `0` there will only be `1` code and thus there will be no resend button. Note that when `Max Retries` is reached, a new code will be generated and sent automatically.

NevisKeyboxProvider_label

Setting the `Label` is required if this pattern is used as a key store provider.

This pattern relies on the standard nevisKeybox mechanism for retrieving the passphrase of the private key.

Run the following commands on all target server(s) to ensure the passphrase can be retrieved:

```
neviskeybox passwd -slot <slot> -label <label> -keep
```

```
neviskeybox access -slot <slot> -label <label> -group nvbgroup
```

The last command will generate a shell script `/var/opt/neviskeybox/default/<slot>/<label>_keypass` which can be invoked by NEVIS components to retrieve the passphrase.

Due to a limitation in some NEVIS components keypass files which contain base64 encoded passphrases are not supported yet. Replace any of the following content with a simple echo returning the passphrase directly.

```
echo "cGFzc3dvcmQ=" | openssl base64 -d
```

nevisKeybox may also be integrated with nevisCred to store the passphrase in a secure place. In this case the shell script will not contain the passphrase but a call of nevisCred.

NevisIDMDeployable_mailSMTPPass

Set if a password is required to connect to the SMTP server.

OAuth2Client_phone

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

CustomRiskScoreWeightConfiguration_locationWeight

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

4.20.0

Full changelog:

Patterns 4.20.0 Release Notes - 2023-08-16

Automatic Key Management (Classic)

In a *classic* (VM) deployment, the automatic key management now generates most of the key material at generation time.

Only JKS and PKCS12 format files are still assembled on the target hosts by running a command. JKS and PKCS12 files are therefore **not** shown in the deployment preview.

Generated key material is stored in the nevisAdmin 4 database until expiration.

The new implementation is simpler and more reliable, but leads to changes in the deployment preview. You can ignore any differences to the `/var/opt/keys` folder.

The automatic key management for classic deployment is still **only** supposed to speed up integration work, but **not** intended to be used in production.

ModSecurity Core Rule Set Upgrade

The default *OWASP ModSecurity Core Rule Set* (CRS) version has been upgraded from `3.3.4` to `3.3.5`.

If you have explicitly selected the `3.3.4` version rule set (`ModSecurity Rule Set` in the `Virtual Host` pattern), you need to change to the new default version `3.3.5`.

nevisAdapt Feedback Configuration change

Feedback-related configuration values from `nevisAdapt Deployable` and `nevisAdapt Authentication Connector` were grouped together in the new `nevisAdapt Feedback Configuration` pattern. Please add one under `nevisAdapt Deployable` / `Advanced Settings` and fill it out with the values from the earlier configuration.

GenericIngressSettings_propagateClientCert

Indicates if the received certificates should be passed on to nevisProxy in the header `ssl-client-cert`.

NevisDetectDatabase_password

Enter the password of the DB connection user.

NevisAuthRealmBase_authConnectorParams

Add custom `init-param` elements to the `Esauth4ConnectorServlet` generated by this pattern.

That servlet is called `Connector_<name>`.

Multi-line values, as required for conditional configuration, can be entered by replacing the line-breaks with `\n`.

Examples:

Key	Value
EnablePollTerminatedCalls	true

NevisIDMJmsQueues_provisioning

NevisIDM JMS Queue to which Provisioning messages should be sent.

Only accepts URIs starting with `amqp`, `amqps` or `Endpoint=sb`. Validates only URIs with `amqp` or `amqps` schemes.

GenericAuthenticationStep_parameters

Define *Template Parameters*.

The syntax is a multi-line String containing a YAML map (key-value pairs). Example:

```
smtp: smtp.siven.ch
sender: noreply@siven.ch
doctype: "&lt;!DOCTYPE html&gt;"
counter: 1
```

As shown in the example above, double quotes `"` need to be put around the value if the value contains special characters.

Parameters can be used in:

- `AuthState(s): direct input`

- `AuthState(s): as file`

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 `\}` .

CustomNevisMetaLogFile_serverSyslogFormat

Log4j 2 log format for the SERVER SYS logs.

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

GroovyScriptStep_onSuccess

Assign an authentication step which shall be executed when the Groovy script sets the result `ok` .

```
response.setResult('ok')
```

If no step is assigned a default state will be added.