If an SP issues a request specifying use of the REFEDS MFA Profile, what can it expect from an IdP that does not support the Profile?

A SAML- or OIDC-compliant IdP/OP MUST treat any unrecognised requested context class or acr in the same way that it would treat one that was recognized but impossible to satisfy. Thus, the response, if issued, would contain the same error status noted in the relevant sections of the Profile.

That said, many software implementations are not in fact compliant and cannot always be trusted to behave according to the letter of the specifications. Ultimately, federations must rely on their members to follow standards, and where necessary, report violations of them.

If an SP requests MFA using the REFEDS MFA Profile, does that alone guarantee MFA will happen?

No.

Most IdPs and campuses that support MFA services do not provide universal MFA coverage for their user communities. This means that even when a given IdP is capable of supporting the Profile, there is some probability that any given user may not be able to authenticate using MFA.

There is no defined mechanism at present to identify whether a given IdP is configured to support the profile, and neither SAML nor OIDC relies on that knowledge; it assumes that IdPs will respond in accordance with the standard when handling a request containing requirements it cannot meet. If the SP does not have any information about an IdPs capabilities, it may not be able to distinguish between a case of specific users being unable to satisfy the profile, and an IdP as a whole not supporting it. Whether this distinction is relevant will depend on the SP.

The information returned in a SAML assertion or ID token is all that one can ultimately depend on and evaluate. It is not sufficient to configure an SP to request MFA and assume all non-error responses will therefore signal the use of MFA. This is because users can generally bypass an SP's SAML or OIDC request configuration using unsolicited responses from an IdP, or by handcrafting a request that does not include the MFA requirement.

It is therefore essential that an SP requiring MFA MUST verify the presence of the value in the incoming assertion or token before proceeding; merely requesting the value does not ensure that the result will contain the proper value even if an assertion or token is returned, for a variety of reasons that are outside the scope of this FAQ. But you can assume that if such a thing were to happen, then either the IdP is not compliant, or that the request it saw was not the request you issued to it. Neither case is under the control of the SP.

An SP prefers MFA, but does not want to require it. What should it do?

Unfortunately, SAML 2.0 does not support this use case directly. It simply does not have the concept of "prefer but not require". While the list of context classes in a request is ordered (in fact this was clarified via errata), this is only useful when different types of authentication are not "composed" together, which is a very common situation with MFA deployments. MFA tends to combine password authentication with "other things" and so it is impossible to reliably ask for "MFA or Password" because one step happens first, satisfies the request, and may well short-circuit any further work in the interest of minimising the user's burden.

Ordering can be useful when methods are discrete, such as Password vs. X.509 certificates vs. Kerberos /SPNEGO. In such a case, the IdP can be expected to try methods in order when it can, but even here IdPs may "prioritise" SSO so that a later method the user has already completed would be favoured over an earlier method that would require a new challenge.

OIDC does have a "voluntary claim" feature that could in theory accommodate this use case, but in the interest of consistency and because of experience with the difficulty of implementing such a feature, the Profile does not specify its use.

Nevertheless, some SPs prefer that a user authenticate with MFA but is willing to accept lesser approaches. Some scenarios where this might make sense:

- Applications that can implement a local scheme to do "stronger authentication" of specific users but prefer to allow users to use familiar campus mechanisms when available.
- Applications that will allow access to some services to all users, but have other services that are limited to those that authenticate using MFA.
Applications that wish to offer their own opt-in feature for users to elect to use MFA for that service.

An application that only allows access to users who authenticate with MFA, but wants to personalise error messages to users who do not use MFA as part of the authentication process.

Because this is not directly supported, there are a couple of alternative strategies to consider for these use cases, though neither is simple to implement.

Separate login actions

A very simplistic approach would be to provide the user with two different login buttons. E.g., “Login with with my institutions SSO and MFA” and “Login with SSO (but no MFA)”, and use that to drive what <AuthnContextClassRef> or acr to request.

“Step Up” MFA

The step-up model is often the most elegant if it can be orchestrated, and is particularly well-suited to cases where either a minority of access needs MFA or where the use of MFA can naturally be required later in the interaction with the user.

With this model, the first interaction with the SP should be handled with a “vanilla” authentication request that does not stipulate anything regarding MFA and simply attempts to get the user identified and into the application.

Assuming this works, the application can proceed for some set of activities (or not), but eventually may either prompt the user to indicate whether MFA is possible, or just unilaterally attempt to elevate the authentication of the user by triggering a second authentication request back to the IdP with the requirement for MFA as described previously. Upon receipt of the new assertion or token, either the user's privileges can be elevated or in the event of an error, left as is with an indication to the user that the request for MFA was not successful.

Again, this may be very non-trivial with some application and SP software or integration patterns.

Note that while many IdPs and many MFA deployments are architected such that the second request for MFA may skip the password step and make for a relatively seamless step-up experience, this is by no means a guarantee. Some IdPs may not have the ability to do this, and some MFA technologies may rely on factors that don’t include the same password.

MFA with Retry

Another common approach is to initially attempt authentication via MFA, but then handle an error from the IdP by downgrading and reissuing a request for regular authentication. In this scenario, the first request is issued with the additional content noted above, but if the IdP responds with an error, the SP has to detect this and either immediately reissue a request without the additional content, or present the user with an explanation and perhaps provide the option to do so.

This can be a bit more interoperable and result in fewer challenges to the user in some cases, but also requires more advanced error handling by the SP and application, so may not always be possible.