PushMDQ implementation guidance for federations

Introduction

The distribution of metadata is the key link that binds together identity federation both at the national and international (inter-federation) level with eduGAIN. As the scale of federations and inter-federations (eduGAIN membership) increases the dominant mechanism for sharing metadata, based on the exchange of a flat file structure becomes problematic. Firstly from the size of the file itself, and secondly from the associated processing delays and distribution of any updates notifying entities of changes. This can have a potentially serious effect in the case of compromise, where a considerable propagation delay can potentially leave a whole federation vulnerable until it is detected. Note that this propagation delay issue is principally relevant to both full mesh federations (which are the bulk of R&E federations) as well as hub and spoke federation on the inter-federation level, though the latter may have alternate distribution methods on a national level.

These issues can be ameliorated by the adoption of the more recent MDQ protocol which allows for the query of metadata relating to one or more entities in a federation, thus resulting in smaller amount of data transfer and potentially smaller propagation delays. However, this approach suffers from both being a single point of failure (for the querying entity) and from adopting a ‘pull-based’ model where the query is subject to deterministic scheduling and thus cannot discover changes in real-time.

To address these issues a new mechanism -PushMDQ- has been proposed. The PushMDQ model leverages existing metadata exchange components, but enhances these with the ability to inform stakeholders about changes in almost real time fashion.
The model is based on a publisher - subscriber model known as websub (formerly PubSubHubbub). Websub was originally intended for data feeds such as RSS or ATOM, but can be applied more generally.
Briefly, a publisher of information (in this case metadata) publishes information to a hub to which interested parties can subscribe to and receive notifications from when new content is available. Hence a subscriber (in this case a federation, or an entity within a federation) can receive instant notification when an update to metadata has occurred and this can (potentially) be propagated across the federation (or federations in the case of eduGAIN) with minimal delay.
We’ve now explained the PUSH, part of PushMDQ, so what about the MDQ part?
With the ability to instantly notify stakeholders of changes, we now gain the ability to also instantly update entity information in the federation metadata. While the proposal is taking into account the flat file metadata exchange, and is believed to be compatible with that model for backwards compatibility, another big improvement can be made by broad adoption

of MDQ technology by federations. The use of MDQ would enable the almost instant publication of new entity information into the metadata more easily and will also enable much smaller, and hence faster updates to entity metadata.

There are a number of benefits to such a system:

- The model does not require modifications to the existing trust model in R&E federations.
- Changing from deterministic pull to a push mechanism reduces pulls by up to 95%;
- Updates can be propagated to all entities instantly (for practical purposes);
- Combining the use of MDQ with PushMDQ reduces the processing overhead even further;
- Backwards compatibility with existing flat-file distribution is possible;
- There is no longer a single point of failure;
- Websub hubs do not need to be intelligent and may exist outside the federation.

However, to maximise the benefits a number of operational and implementation best practices should be implemented. These could be staged over time in order to ensure a smooth transition and minimise disruption and burden on federation resources. By doing so incremental improvements could be realised until full adoption can be accomplished.

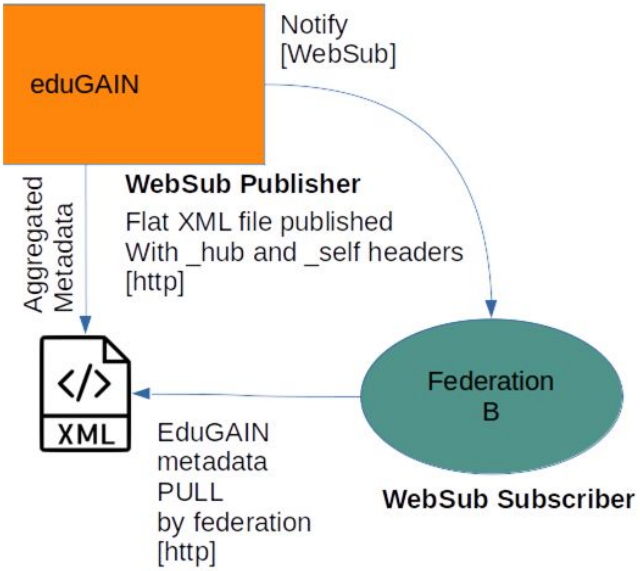
This overview describes a proposal to the implementation steps to move towards an adoption of the pushMDQ model for metadata exchange.

Operational and implementation guidance

Use of flat file structure and MDQ

The aim of the use of PushMDQ protocol within eduGAIN is to reduce the time for an entity change in the federation metadata to propagate from the originating federation to all federations within eduGAIN. At the same time the current proposal seeks to maintain backwards compatibility with the bulk of federations that currently use a flat file exchange rather than MDQ (thus allowing the possibility of a per entity exchange. In general the propagation time is the result of a number of factors:

1. The time taken for a federation to become aware that a change has taken place;
2. The time taken for batch processing and publication of the XML metadata
3. The time at which batch processing is scheduled
4. The time taken for a federation to update its cache based on cache-duration. Note: This can be accelerated to some extent by using an appropriate http conditional request (precondition header).
5. The expiration of the metadata as given by the ValidUntil attribute.



Using the websub mechanism of PushMDQ the notification (1) from the eduGAIN hub is effectively instantaneous. However the subscribing federation wishing to obtain the updated XML metadata needs to wait for the publishing federation to process the xml data from it's entities (2) and publish this to eduGAIN hub for aggregation, which is generally done at specific times (3). Hence continued use of a flat file structure to propagate XML metadata changes removes some of the benefits of using PushMDQ, although since federation 'pulls' of the flat file are driven by websub notifications, rather than cache duration timeouts this still improves efficiency by eliminating redundant 'uniformed pulls' and moving to a 'notification pull' model. The maximum benefit of using PushMDQ whilst still using a flat file structure depends on the batch processing delays of other federations within eduGAIN.

To fully maximise the benefits of using PushMDQ the subscribing federation should ideally adopt the use of MDQ in order to query the changes related to specific entities.

In this case the publishing federation needs only to inform the eduGAIN hub of changed entities whose metadata can then be queried using MDQ and aggregated and the subscribing federation (once notified) is able to query specific changes using MDQ. This greatly reduces the processing delay by the publishing federation, and to some extent the processing delay at the subscriber (although this is less significant). Although, even in this case the total processing time is dependent on the MDQ batch processing by the publishing entity and in general federations don't pull and process all metadata when notified of a single change.
Best performance will be obtained when both publishing and subscribing entities use MDQ.

Processing of change notifications

To minimise the processing time, publishing federations should pull all changes (whether they aggregate a flat file or MDQ) when a threshold of change is notified from its federated entities. This threshold needs to be set such that there is a balance between the improved propagation time of having updated metadata published frequently by the federation and the increased processing related to a higher number of pulls.
Similarly subscribing federations should always pull metadata (either flat file or MDQ) when an eduGAIN hub notification is received. Note: In general federations are both publishers and subscribers.
The ultimate solution to this problem is to extend the PushMDQ protocol down within federations to the individual entities, but this would require a change to every SP and IdP and so is something that would need to be phased in over time.
In addition a similar benefit could be had when the federation would publish its metadata via MDQ where the updated frequency as well as the cache timeout of the entities in the federation would be low

Trust mechanism for endpoints

The trust model for endpoints, either entities or federations, does not change with PushMDQ proposal as at no point in time the Websub part of PushMDQ publishes actual updates on entities. PushMQD only publishes information on the fact a given entity has changed, and refers to the relevant metadata endpoint to learn about the update.
There are however some technical ways to implement better security in the message exchange between hub and subscriber described in the protocol, which seem logical additions, and hence are recommended.

Resilience

Since a hub can fail and this will result in notifications potentially being missed or received out of order by subscribers a mechanism is needed in order to provide resilience and handle the case where multiple hubs are sending notifications.
As a first precaution, many federation policies already dictate a minimal refresh time for the metadata, like e.g. 6 hours for eduGAIN. Federations and entities should keep such minimal refresh criteria and use them to update in case no notification was received at all within the agreed timeframe. This would mitigate a total failure of a hub, and in fact represent the current state of affairs without PushMDQ.

On top of that a number of technical measure can be taken to:

- A subscriber may subscribe to multiple hubs and where this is used to improve resilience a synchronisation mechanism needs to be in place so the data served by the hubs is identical. This can be accomplished by a combination of database virtualization, clustering and mirroring.
- **Federations should consider the provision of multiple hubs that provide the same synchronized notifications or a single hub with a failover mechanism (the exact mechanism needs more study and may vary between federations depending on their existing architecture).**
- Root publishers should add a timestamp or some other form of sequence number when notifying the hub of the availability of new content. This can then be used by the subscriber in the notification callback message to ensure that content is not delivered out of order.

Hub notification time out and back-off mechanism

The subscriber should respond to metadata update notifications from the hub as quickly as possible. Where this does not happen (for example) if the subscriber is busy with other processing) the hub will retry the update request for a given period before giving up. This could cause large quantities of update requests to occur where the subscriber cannot respond quickly enough to prevent the hub backing-off and retrying the content distribution request.

Federations could ensure that publishers do not send entity notifications after an aggregated entity notification, if there have been no changes. (this needs further consideration).
It is recommended the subscribers implement a proper mechanism to cache update requests so response to the hub can be done in a timely fashion.