Authentic 2 - Development #28215

permettre le filtrage des frontends d'authentification à exposer à l'usager

21 novembre 2018 11:58 - Serghei Mihai

Statut:	Fermé	**Début:**	21 novembre 2018
Priorité:	Normal	**Echéance:**	
Assigné à:	Serghei Mihai	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Le(s) filtre(s) serai-en-t une expression python/django à placer dans les settings du tenant.

Demandes liées:

Lié à Authentic 2 - Development #28216: déclenchement automatique du SSO si u...	**Fermé**	**21 novembre 2018**
Lié à Authentic 2 - Development #30960: Permetttre de faire d'authentic un Id...	**Fermé**	**27 février 2019 31 mai 2019**
Lié à Authentic 2 - Bug #31259: présenter les IDP OpenidConnect dans des bloc...	**Fermé**	**11 mars 2019**
Lié à Authentic 2 - Development #31218: registration: permettre au frontend L...	**Fermé**	**08 mars 2019**

Révisions associées

Révision 3043ff34 - 13 mars 2020 10:21 - Serghei Mihai

misc: allow authenticators display conditions (#28215)

Historique

#2 - 21 novembre 2018 12:03 - Serghei Mihai

- Lié à Development #28216: déclenchement automatique du SSO si un seul frontend est disponible ajouté

#3 - 22 novembre 2018 15:51 - Serghei Mihai

- Fichier 0001-misc-add-support-for-request-based-enable-conditions.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Basé sur la branche du ticket #14475 (renommage des classes et modules d'authentification).

Il manque pas mal de tests et peut-être que la terminologie n'est pas la meilleure, mais l'idée est de pouvoir une simple expression pythonique pour activer chaque frontend d'authentification.
Il manque encore des tests: wip.

#4 - 22 novembre 2018 16:36 - Frédéric Péters

À reculer un peu,

- on aurait des conditions "wants_frontoffice" / "wants_backoffice"
- genre une configuration pourrait être : FranceConnect:wants_frontoffice, KeyCloak:wants_backoffice.
  - l'usager arrive, les filtres sont joués, reste une seule méthode, l'autorun de #28216 est lancé, etc.

Mais, la situation GNM est différente, il y a une seule méthode, "oidc", mais celle-ci en cache deux (oidc vers GLC, oidc vers GLC/agents). Résultat cette mécanique de filtre ne peut pas y fonctionner.

(c'est un commentaire que j'ai fait dans les notes "bilan J1" du pad, sans doute j'aurais du la mettre en avant).

Selon moi, on a besoin d'une découpe supplémentaire, un module d'authentification (OIDC, FC, etc.) doit pouvoir exposer plusieurs frontends(?), et c'est à ce niveau que pourrait alors se jouer le filtrage de ce ticket.

- OIDC (module d'authent)
  - explode
  - → frontends : OIDC[usagers], OIDC[agents]
  - filtre des frontends
  - → OIDC[agents]
    - un seul → autorun

#5 - 22 novembre 2018 16:50 - Benjamin Dauvergne

Frédéric Péters a écrit :

> À reculer un peu,
>
> - on aurait des conditions "wants_frontoffice" / "wants_backoffice"
> - genre une configuration pourrait être : FranceConnect:wants_frontoffice, KeyCloak:wants_backoffice.
>   - l'usager arrive, les filtres sont joués, reste une seule méthode, l'autorun de #28216 est lancé, etc.
>
> Mais, la situation GNM est différente, il y a une seule méthode, "oidc", mais celle-ci en cache deux (oidc vers GLC, oidc vers GLC/agents). Résultat cette mécanique de filtre ne peut pas y fonctionner.

Si parce que pour le frontend OIDC, chaque fournisseur OIDC représente une méthode différente.

> (c'est un commentaire que j'ai fait dans les notes "bilan J1" du pad, sans doute j'aurais du la mettre en avant).
>
> Selon moi, on a besoin d'une découpe supplémentaire, un module d'authentification (OIDC, FC, etc.) doit pouvoir exposer plusieurs frontends(?), et c'est à ce niveau que pourrait alors se jouer le filtrage de ce ticket.
>
> - OIDC (module d'authent)
>   - explode
>   - → frontends : OIDC[usagers], OIDC[agents]
>   - filtre des frontends
>   - → OIDC[agents]
>   - un seul → autorun

Dans mon plan initial ce n'est pas la vue login() qui décide de lancer autorun, mais le frontend il faut pour cela deux méthodes (can_autorun() puis autorun()) ou bien prévoir qu'autorun retourne possiblement None.

Donc dans un fonctionnement avec None:

- GNM -> un seul frontend = OIDC
- appel à OIDCFrontend.autorun()
  - sélection d'au moins deux fournisseurs : autorun() retourne None
  - sélection d'un seul fournisseur : autorun() retourne HttpRedirect

Au passage il faut passer à autorun tout un tas de paramètre reçus par login, comme le nonce, next_url, etc..

#6 - 22 novembre 2018 16:57 - Frédéric Péters

> Si parce que pour le frontend OIDC, chaque fournisseur OIDC représente une méthode différente.

C'est peut-être un problème de vocabulaire mais il me semble que c'est ma proposition aussi, et qu'elle ne correspond pas à la série de patchs en cours de développement, #14475 change juste des noms, on reste avec OIDC = 1 Frontend^WAuthenticator, et ce ticket, il itère sur ces objets, pas un objet "méthode d'authentification" dessous.

#7 - 22 novembre 2018 17:01 - Serghei Mihai

Frédéric Péters a écrit :

> > Si parce que pour le frontend OIDC, chaque fournisseur OIDC représente une méthode différente.
>
> C'est peut-être un problème de vocabulaire mais il me semble que c'est ma proposition aussi.

Oui. Je revois mon idée pour prendre en compte ce cas d'usage.

#8 - 03 décembre 2018 10:47 - Benjamin Dauvergne

Si on introduit maintenant un changement de structure avec des AuthMethod il faut passer sur authentic2-auth-fc aussi et peut-être d'autres modules externes, je préfèrerai que ce soit fait une fois ces modules externes réintégrés dans authnetic.

#9 - 03 décembre 2018 11:18 - Benjamin Dauvergne

- Statut changé de Solution proposée à Nouveau

#10 - 13 décembre 2018 09:13 - Frédéric Péters

(peut-être que tout ceci devrait être dans authentic, pas authentik)

À redéfinir le fonctionnement des frontends/méthodes d'authentification il faut peut-être inclure dans la réflexion également la partie "inscription" (qui aujourd'hui mélange un truc un peu en dur pour l'inscription locale/email et un mécanique "accumulation de blocs de frontends" simulaire à la page de login).

#11 - 13 décembre 2018 11:24 - Benjamin Dauvergne

- Projet changé de Authentik à Authentic 2

Œuf corse.

#12 - 01 février 2019 16:55 - Serghei Mihai

- Fichier 0002-misc-add-support-for-request-based-enable-conditions.patch ajouté

- Fichier 0001-auth-let-LoginPassowrd-frontend-handle-registration-.patch ajouté

- Tracker changé de Support à Bug

- Statut changé de Nouveau à Solution proposée

Patch en plus pour ne pas mettre en dur le formulaire de création de compte sur la page d'enregistrement.

Je ne suis pas sûr de l'approche d'appeler l'autorun directement depuis le vue login j'aimerais qu'on puisse aussi laisser la main à l'usager de choisir son moyen d'authentification.
Je pense notamment au cas de 3 frontends: login/mdp, FranceConnect et un IDP dédié uniquement aux agents. Pour l'agent on peut souhaiter de jouer l'autorun immédiatement, mais l'usager doit pouvoir choisir.

Concernant la problèmatique des multiples fournisseurs OIDC je vois la possibilité de pouvoir les filtrer dans la condition d'affichage, genre: request.session.get('is_agent', False) and providers.filter(slug="idp-agents")

#13 - 03 février 2019 17:08 - Frédéric Péters

> Concernant la problèmatique des multiples fournisseurs OIDC je vois la possibilité de pouvoir les filtrer dans la condition d'affichage, genre: request.session.get('is_agent', False) and providers.filter(slug="idp-agents")

Je trouve vraiment dommage de garder cette association "1 module d'authentification = 1 pavé de connexion", qui oblige derrière à inventer une notion de sous-pavé (un fournisseur oidc), qui fait que tu parles déjà de devoir dupliquer une mécanique de conditions. Je ne fais que me répéter.

#14 - 27 février 2019 16:42 - Mikaël Ates (de retour le 29 avril)

- Lié à Development #30960: Permetttre de faire d'authentic un IdP proxy transparent ajouté

#16 - 26 novembre 2019 15:41 - Serghei Mihai

- Lié à Bug #31259: présenter les IDP OpenidConnect dans des blocks à part sur la page de login ajouté

#17 - 26 novembre 2019 16:37 - Serghei Mihai

- Lié à Development #31218: registration: permettre au frontend LoginPassword de gérer la création du cmpte ajouté

#18 - 03 décembre 2019 18:17 - Serghei Mihai

- Fichier 0001-misc-add-support-for-enable-conditions-on-authentica.patch ajouté

En reprenant le principe de plus haut, avec possibilité de filtrer aussi les idp OIDC.

#19 - 03 décembre 2019 18:45 - Frédéric Péters

```
    A2_AUTH_PASSWORD_ENABLE_CONDITION=Setting(default=None, definition='Filters',
                                              names=('FRONTEND ENABLE CONDITION',)),
```

Je lis le code de app_settings et names me semble être un attribut permettant d'aller chercher, si le paramètre demandé n'existe pas, le paramètre sous un autre nom. Et ici, sous le nom "FRONTEND ENABLE CONDITION", avec des espaces, ça me semble farfelu.

```
                return eval(condition, context)
```

Discussion de fond, je serais pour éviter le Python tapable partout, que se passe par une condition Django. (mais je pense qu'il n'y aura pas accord sur ce commentaire, ne te lance surtout pas là-dedans tout de suite).

```
            except Exception, e:
```

Pensons à Python 3, Exception as e.

```
            providers = self.eval_enable_condition(app_settings.IDP_ENABLE_CONDITION, request,
                                                   providers=providers)
```

mais self.eval_enable_condition retourne un booléen. (?)

Ici, et dans l'autre, côté SAML, où on passe idps=idps, je ne pige pas trop le sens de l'affaire, pourquoi on ferait une comparaison sur base de la liste des IdP possibles. Alors ça s'éclaire en lisant le test, providers.filter(name='My IDP'), mais pour moi c'est retourner à mon commentaire 13 ci-dessus, un partage bizarre d'une même mécanique pour faire deux choses :

- pour une méthode d'authentification comme le mot de passe avoir une condition qui retourne un booléen.
- pour une méthode d'authentification comme oidc/saml, avoir une condition qui n'en est pas une qui est plutôt une expression filtrant une liste.

Pas fan du tout il y a dix mois, pas fan du tout aujourd'hui.

Très concrètement, pour mon usage GNM, j'aurais aimé pouvoir :

```
"A2_AUTH_MODULES_CONDITIONS": {
  "password": "request|is_for_backoffice",
  "oidc/glc": "not request|is_for_backoffice"
}
```

ou dans la version précédente avec deux GLC,

```
"A2_AUTH_MODULES_CONDITIONS": {
  "oidc/glc-agents": "request|is_for_backoffice",
  "oidc/glc": "not request|is_for_backoffice"
}
```

Ailleurs, pour un Chambéry,

```
"A2_AUTH_MODULES_CONDITIONS": {
  "password": "not request|is_for_backoffice",
  "franceconnect": "not request|is_for_backoffice",
  "saml/adfs": "request|is_for_backoffice"
}
```

(et j'en reste ici à des settings, idéalement j'aimerais la condition portée par l'objet OIDCProvider, etc.)

#20 - 19 décembre 2019 18:12 - Serghei Mihai

- Fichier 0001-misc-add-support-for-enable-conditions-on-authentica.patch ajouté

Patch refait pour aller dans le sens de tes remarques.
Et is_for_backoffice regarde dans la session si un flag, posé quand mellon nous passe son login-hint, est présent.

#21 - 03 janvier 2020 11:59 - Brice Mallet

- Tracker changé de Bug à Development

#22 - 14 janvier 2020 09:44 - Serghei Mihai

Je veux bien une relecture du dernier patch.

#23 - 16 janvier 2020 16:20 - Serghei Mihai

- Fichier 0001-misc-add-support-for-enable-conditions-on-authentica.patch ajouté

Thomas m'a rappelé à l'instant qu'on pouvait aussi filtrer les authenticators SAML vu qu'on les distingue maintenant sur la page de login.
J'ai rajouté ça.

#24 - 21 janvier 2020 23:05 - Benjamin Dauvergne

- Statut changé de Solution proposée à En cours

Je ne peux pas accepter de conditions basés sur des templates Django ; ou alors faut trouver un autre développeur.

PS: il y a tout ce qu'il faut là, http://git.entrouvert.org/authentic.git/tree/src/authentic2/utils/evaluate.py c'était prévu pour ce genre d'usage.

#25 - 22 janvier 2020 10:15 - Serghei Mihai

Ok. J'avais pensé au début à l'usage de utils/evaluate mais l'idée d'uniformiser la syntaxe des expressions en Django me plasait bien aussi.

Pour revenir à l'écriture des expressions de filtrage, on reste sur la définition des celles-ci dans le settings.json du tenant comme proposé plus haut, genre ?:

```
"A2_AUTH_MODULES_CONDITIONS": {
  "password": "is_for_backoffice(request)",
  "franceconnect": "is_for_backoffice(request)",
  "saml/adfs": "is_for_backoffice(request)"
}
```

#26 - 26 janvier 2020 19:06 - Benjamin Dauvergne

Serghei Mihai a écrit :

> Pour revenir à l'écriture des expressions de filtrage, on reste sur la définition des celles-ci dans le settings.json du tenant comme proposé plus haut, genre ?:
> [...]

1. Idéalement on va vouloir ça par instance, donc non pas comme cela, déjà on ne va pas vouloir que ça accède à request il faut un truc plus contraignant que cela et je ne mettrai pas ça au niveau de get_backends() plutôt dans login() ; je veux vraiment éviter des expressions du style request.GET.get('idee-du-jour') == 'ok',
2. ne pas mélanger les histoires de login-hint SAML on verra ça dans un autre ticket.

Je propose de faire simple pour l'instant :

- dans login on filtre authenticators ou les instances sur l'absence d'une fonction shown qui a pour signature shown(insance_id, **kwargs) ou alors sur le résultat de son appel,
- dans kwargs dans un premier temps on peut mettre des trucs évidents genre remote_addr, service_id

On verra plus tard quoi faire de login_hint et comment le passer, mais la mécanique étant faite ce sera plus simple.

PS: on a déjà dans get_backends() de quoi passer des paramètre à Authenticator.__init__, donc une première implémentation de base tirerai l'expression ce ça, genre

```
class BaseAuthenticator:
    def __init__(self, shown_condition=None):
        self.shown_condition = shown_condition

    def get_shown_condition(self, instance_id=None, instance=None):
        if isinstance(self.shown_condition, dict):
            if instance_id and instance_id in self.shown_condition:
                return self.shown_condition[instance_id]
            else:
                return self.shown_condition.get('')
        else:
            return self.shown_condition

    def shown(self, instance_id=None, instance=None, remote_addr=None, service_id=None):
        condition = self.get_shown_condition(instance_id=instance_id, instance=None)
        if not condition:
            return True
        return eval(condition, {'id': instance_id, 'remote_addr': remote_addr, 'service_id': service_id})
```

Ça se configure avec :

```
A2_AUTH_FRONTENDS_KWARGS = {
    'password': {
        'shown_condition': '\'admin\' in login_hint',
    },
    'oidc': {
        'shown_condition': '\'admin\' not in login_hint',
    }
}
```

coté auth_oidc on pourra avoir

```
    def get_shown_condition(self, instance_id, instance):
         if instance.shown_condition:
             return instance.shown_condition
         return super(OIDCAuthenticator, self).get_shown_condition(instance_id=instance_id, instance=instance)
```

avec ajout d'un champ au modèle.

#27 - 09 mars 2020 16:52 - Serghei Mihai

Benjamin Dauvergne a écrit :

> coté auth_oidc on pourra avoir
>
> [...]
>
> avec ajout d'un champ au modèle.

Je pense qu'il faut soit rajouter un champ au modèle et gérer l'affichage uniquement sur la base celui-ci, soit le faire uniquement dans le settings, genre:

```
{
 "oidc": {
  "show_condition: {
    "idp-agents": "<condition>",
    "idp-usagers": "<condition>"
  }
}
```

pour éviter d'avoir plusieurs mécanismes d'affichage.

Pareil pour auth_saml. Soit:

```
{
 "saml": {
  "show_condition": {
    "idp_ville": "<condition>",
    "idp_departement": "<condition>"
  }
}
```

ou "idp_ville", "idp_departement" sont les slugs des MELLON_IDENTITY_PROVIDERS.
Oubien définir les clés SHOW_CONDITION dans MELLON_IDENTITY_PROVIDERS.

Et donc avoir une seule manière de gérer l'affichage des backends proposant plusieurs instances.

#28 - 10 mars 2020 16:45 - Serghei Mihai

- Fichier 0001-misc-allow-authenticators-display-conditions-28215.patch ajouté

- Statut changé de En cours à Solution proposée

Serghei Mihai a écrit :

> pour éviter d'avoir plusieurs mécanismes d'affichage.
>
> Et donc avoir une seule manière de gérer l'affichage des backends proposant plusieurs instances.

Patch allant dans ce sens.

#29 - 12 mars 2020 12:12 - Benjamin Dauvergne

- je ne mélangerai pas enabled et shown pour l'instant (1 parce que je ne vois pas à quoi ça sert là maintenant, 2 parce que dans les cas d'authentification déporté ça peut avoir un sens justement, typiquement OIDC ou SAML on peut se connecter en utilisant une truc initié par l'IdP ou une URL d'initiation sur authentic sans passer par un bouton sur le front).

Sinon pour le reste c'est ok pour moi.

#30 - 12 mars 2020 14:20 - Serghei Mihai

- Fichier 0001-misc-allow-authenticators-display-conditions-28215.patch ajouté

Tu as raison.

#31 - 12 mars 2020 19:31 - Benjamin Dauvergne

Serghei Mihai a écrit :

> Tu as raison.

Je préfère un db explicite.

```
pytestmark = pytest.mark.django_db
```

#32 - 12 mars 2020 23:23 - Serghei Mihai

- Fichier 0001-misc-allow-authenticators-display-conditions-28215.patch ajouté

Voilà.

#33 - 13 mars 2020 10:04 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

#34 - 13 mars 2020 10:24 - Serghei Mihai

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 3043ff34217f6026640c2f27791e4813d9d5f207 (origin/master, origin/HEAD)
Author: Serghei Mihai <smihai@entrouvert.com>
Date:   Mon Mar 9 15:19:47 2020 +0100

    misc: allow authenticators display conditions (#28215)
```

#35 - 19 mars 2020 22:14 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-misc-add-support-for-request-based-enable-conditions.patch	10,5 ko	22 novembre 2018	Serghei Mihai
0002-misc-add-support-for-request-based-enable-conditions.patch	11,7 ko	01 février 2019	Serghei Mihai
0001-auth-let-LoginPassowrd-frontend-handle-registration-.patch	2,29 ko	01 février 2019	Serghei Mihai
0001-misc-add-support-for-enable-conditions-on-authentica.patch	12,4 ko	03 décembre 2019	Serghei Mihai
0001-misc-add-support-for-enable-conditions-on-authentica.patch	15,6 ko	19 décembre 2019	Serghei Mihai
0001-misc-add-support-for-enable-conditions-on-authentica.patch	18,6 ko	16 janvier 2020	Serghei Mihai
0001-misc-allow-authenticators-display-conditions-28215.patch	16,2 ko	10 mars 2020	Serghei Mihai
0001-misc-allow-authenticators-display-conditions-28215.patch	16,3 ko	12 mars 2020	Serghei Mihai
0001-misc-allow-authenticators-display-conditions-28215.patch	16,3 ko	12 mars 2020	Serghei Mihai