

0936ea2533 ▾

...

django-mfa2 / mfa / FIDO2.py / <> Jump to ▾



mkalioby Merged v2.5 ✓

History

1 contributor

173 lines (152 sloc) | 7.09 KB

...

```
1 from fido2.client import Fido2Client
2 from fido2.server import Fido2Server, PublicKeyCredentialRpEntity
3 from fido2.webauthn import AttestationObject, AuthenticatorData, CollectedClientData
4 from django.template.context_processors import csrf
5 from django.views.decorators.csrf import csrf_exempt
6 from django.shortcuts import render
7 # from django.template.context import RequestContext
8 import simplejson
9 from fido2 import cbor
10 from django.http import HttpResponse
11 from django.conf import settings
12 from .models import *
13 from fido2.utils import websafe_decode, websafe_encode
14 from fido2.webauthn import AttestedCredentialData
15 from .views import login, reset_cookie
16 import datetime
17 from .Common import get_redirect_url
18 from django.utils import timezone
19
20
21 def recheck(request):
22     """Starts FIDO2 recheck"""
23     context = csrf(request)
24     context["mode"] = "recheck"
25     request.session["mfa_recheck"] = True
26     return render(request, "FIDO2/recheck.html", context)
27
28
```

```

29 def getServer():
30     """Get Server Info from settings and returns a Fido2Server"""
31     rp = PublicKeyCredentialRpEntity(id=settings.FIDO_SERVER_ID, name=settings.FIDO_SERVER_NAME)
32     return Fido2Server(rp)
33
34
35 def begin_registration(request):
36     """Starts registering a new FIDO Device, called from API"""
37     server = getServer()
38     registration_data, state = server.register_begin({
39         u'id': request.user.username.encode("utf8"),
40         u'name': (request.user.first_name + " " + request.user.last_name),
41         u'displayName': request.user.username,
42     }, getUserCredentials(request.user.username))
43     request.session['fido_state'] = state
44
45     return HttpResponse(cbor.encode(registration_data), content_type = 'application/octet-stream')
46
47
48 @csrf_exempt
49 def complete_reg(request):
50     """Completes the registration, called by API"""
51     try:
52         data = cbor.decode(request.body)
53
54         client_data = CollectedClientData(data['clientDataJSON'])
55         att_obj = AttestationObject((data['attestationObject']))
56         server = getServer()
57         auth_data = server.register_complete(
58             request.session['fido_state'],
59             client_data,
60             att_obj
61         )
62         encoded = websafe_encode(auth_data.credential_data)
63         uk = User_Keys()
64         uk.username = request.user.username
65         uk.properties = {"device": encoded, "type": att_obj.fmt, }
66         uk.owned_by_enterprise = getattr(settings, "MFA_OWNED_BY_ENTERPRISE", False)
67         uk.key_type = "FIDO2"
68         uk.save()
69         return HttpResponse(simplejson.dumps({'status': 'OK'}))
70     except Exception as exp:
71         import traceback
72         print(traceback.format_exc())
73         try:
74             from raven.contrib.django.raven_compat.models import client
75             client.captureException()
76         except:
77             pass

```

```

78         return HttpResponse(simplejson.dumps({'status': 'ERR', "message": "Error on server, please
79
80
81 def start(request):
82     """Start Registration a new FIDO Token"""
83     context = csrf(request)
84     context.update(get_redirect_url())
85     return render(request, "FIDO2/Add.html", context)
86
87
88 def getUserCredentials(username):
89     credentials = []
90     for uk in User_Keys.objects.filter(username = username, key_type = "FIDO2"):
91         credentials.append(AttestedCredentialData(websafe_decode(uk.properties["device"])))
92     return credentials
93
94
95 def auth(request):
96     context = csrf(request)
97     return render(request, "FIDO2/Auth.html", context)
98
99
100 def authenticate_begin(request):
101     server = getServer()
102     credentials = getUserCredentials(request.session.get("base_username", request.user.username))
103     auth_data, state = server.authenticate_begin(credentials)
104     request.session['fido_state'] = state
105     return HttpResponse(cbor.encode(auth_data), content_type = "application/octet-stream")
106
107
108 @csrf_exempt
109 def authenticate_complete(request):
110     try:
111         credentials = []
112         username = request.session.get("base_username", request.user.username)
113         server = getServer()
114         credentials = getUserCredentials(username)
115         data = cbor.decode(request.body)
116         credential_id = data['credentialId']
117         client_data = CollectedClientData(data['clientDataJSON'])
118         auth_data = AuthenticatorData(data['authenticatorData'])
119         signature = data['signature']
120     try:
121         cred = server.authenticate_complete(
122             request.session.pop('fido_state'),
123             credentials,
124             credential_id,
125             client_data,
126             auth_data,

```

