zephir-client - Anomalie #738

Effet de bord possible lié aux EntLogin

30/06/2010 15:43 - Joël Cuissinat

Statut:	Ne sera pas résolu	**Début:**	30/06/2010
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0.00 heure
Version cible:		**Temps passé:**	0.00 heure
Distribution:	EOLE 2.3		

Description

```
______________________________________________________ entrypoint: test_create_eleve _____________
_________________________________________

    def test_create_eleve():
        backend.ajout_eleve('televe', 'tpass', 'nom', 'prenom', '01021980', 'tclasse', '666',
>                     civilite='1', domaine='restreint', quota='20', profil='3', shell=False)

[/usr/share/eole/tests/test_scribe.py:30]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def ajout_eleve(login, password, nom, prenom, date, classe, numero, civilite='1',
            domaine='restreint', quota='10', profil='1', shell=False):
        """
        ajout d'un élève
        @param login  : login élève
        @param password : mot de passe
        @param nom : nom de famille
        @param prenom : prénom
        @param date : date de naissance (jj/mm/aa[aa] ou jjmmaa[aa])
        @param classe : classe de l'élève
        @param numero : numéro élève
        @param civilite : civilité (1=M., 2=Mme, 3=Mlle)
        @param domaine : domaine mail (restreint ou internet)
        @param quota : quota éléve
        @param profil : profil (1=local, 2=obligatoire-1, 3=obligatoire-2, 4=itinérant)
        @param shell : activation du shell
        """
        if domaine.startswith('i-'):
            domaine = 'restreint'

        eleve = {}
        eleve['login'] = login
        eleve['password'] = password
        eleve['nom'] = nom
        eleve['prenom'] = prenom
        eleve['date'] = date
        eleve['classe'] = classe
        eleve['numero'] = numero
        eleve['civilite'] = civilite
        eleve['domaine'] = domaine
        eleve['quota'] = quota
        eleve['profil'] = profil
        eleve['shell'] = shell

        ldapuser = Eleve()
>       ldapuser.add_one(**eleve)

[/var/lib/python-support/python2.5/scribe/backend.py:605]
```

```
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def add_one(self, **args):
        """
                ajout d'un utilisateur (mode déconnecté)
            """
        self.ldap_admin.connect()
>       self._add(**args)

[/var/lib/python-support/python2.5/scribe/eoleuser.py:249]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def _add(self, **args):
        """
                Ajoute un utilisateur
                **args
            """
        self.filter_must_args(args)
        self.filter_may_args(args)
        if match("^[a-zA-Z0-9.\-_]*$", args['login']) is None:
            raise BadLogin("Login \"%s\" invalide !" % args['login'])
        self._test_available_name(args['login'])
        # FIXME : fixes #327 mais est-ce le bon endroit ?
        if tool.not_empty(args, 'date'):
            args['date'] = tool.deformate_date(args['date'])
        if self.has_samba:
            user_add_args = self.get_smbldap_useradd_args(**args)
            self.exec_smbldap_useradd(user_add_args)
            self.c_mod_password(args['login'], args['password'])
            if args['change_pwd']:
                self.password_must_change(args['login'])

        if self.has_ftp:
            self._gen_ftpdir(args['login'])

        self._add_perso(**args)

>       self._add_scribe_user(**args)

[/var/lib/python-support/python2.5/scribe/eoleuser.py:277]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def _add_scribe_user(self, login, **args):
        """
                ajout les attributs Scribe à un Eleve
            """
        mail = "%s@%s" % (login, MAIL_DOMAIN[args['domaine']])
        maildir = join(HOME_PATH, login[0], login, 'MailDir') + '/'
        args['niveau'] = self._get_niveau(args['classe'])
        datas = []
        user_dn = USER_DN % dict(uid=login, _type=self._type)
        objectclass = self._get_attr(login, 'objectClass')
        objectclass.extend(['Eleves', 'ENTPerson', 'ENTEleve', 'radiusprofile'])
        datas.append((MOD_REPLACE, 'objectClass', objectclass))
>       datas.extend(gen_common_attrs(login, entprofil=self.profil, **args))

[/var/lib/python-support/python2.5/scribe/eleves.py:83]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def gen_common_attrs(login, **args):
        """
        Gestion des attibuts communs
        """
```

```
        attrs = []
        attrs.append((MOD_REPLACE, 'cn', "%(prenom)s %(nom)s" % args ))
        attrs.append((MOD_REPLACE, 'sn', args['nom'] ))
        attrs.append((MOD_REPLACE, 'givenName', args['prenom'] ))
        attrs.append((MOD_REPLACE, 'displayName', "%(prenom)s %(nom)s" % args ))
        attrs.append((MOD_REPLACE, 'gecos', tool.replace_cars("%(prenom)s %(nom)s" % args) ))
        today = tool.format_current_date()
        attrs.append((MOD_REPLACE, 'LastUpdate', today))
        if args['entlogin']:
            # affectation immédiate
>           entlogin = get_one_entlogin()

[/var/lib/python-support/python2.5/scribe/eoleuser.py:94]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def get_one_entlogin():
        """
            récupération d'un nouveau login ENT
        """
>       pool = get_single_id()

[/var/lib/python-support/python2.5/scribe/entlogin.py:17]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def get_single_id(code_ent = None):
        pool = get_pool(code_ent)
        if pool is None:
            return pool
        # essai de récupération d'un identifiant
        if pool.free_space < 1:
            if not registered:
                print_red("""Pas assez d'identifiants disponibles.
    Serveur non enregistré sur Zephir.
    La récupération automatique d'identifiants n'est pas gérée""")
                return None
            else:
                print_orange("""Pas assez d'identifiants disponibles""")
                # calcul du nombre d'identifiants à demander (identifiants manquants + stock minim
um)
>               if get_new_ids(MIN_STOCK):

[/var/lib/python-support/python2.5/zephir/manage_pool.py:134]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def get_new_ids(num_ids):
        # lecture de la cle ssh publique
        cle_pub = file('/var/spool/uucp/.ssh/id_rsa.pub').read().strip()
        try:
>           code, res = convert(zephir_proxy.entid.get_id_range(id_serveur, base64.encodestring(cl
e_pub), num_ids))

[/var/lib/python-support/python2.5/zephir/manage_pool.py:53]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def __call__(self, *args):
>       return self.__send(self.__name, args)

[/usr/lib/python2.5/xmlrpclib.py:1147]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ 
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def __request(self, methodname, params):
        # call a method on the remote server
```

```
        request = dumps(params, methodname, encoding=self.__encoding,
                        allow_none=self.__allow_none)

        response = self.__transport.request(
            self.__host,
            self.__handler,
            request,
>           verbose=self.__verbose

[/usr/lib/python2.5/xmlrpclib.py:1437]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def request(self, host, handler, request_body, verbose=0):
        # issue XML-RPC request

        h = self.make_connection(host)
        if verbose:
            h.set_debuglevel(1)

        self.send_request(h, handler, request_body)
        self.send_host(h, host)
        self.send_user_agent(h)
>       self.send_content(h, request_body)

[/usr/lib/python2.5/xmlrpclib.py:1183]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def send_content(self, connection, request_body):
        connection.putheader("Content-Type", "text/xml")
        connection.putheader("Content-Length", str(len(request_body)))
>       connection.endheaders()

[/usr/lib/python2.5/xmlrpclib.py:1297]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def endheaders(self):
        """Indicate that the last header line has been sent to the server."""

        if self.__state == _CS_REQ_STARTED:
            self.__state = _CS_REQ_SENT
        else:
            raise CannotSendHeader()

>       self._send_output()

[/usr/lib/python2.5/httplib.py:860]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def _send_output(self):
        """Send the currently buffered request and clear the buffer.

            Appends an extra \\r\\n to the buffer.
            """
        self._buffer.extend(("", ""))
        msg = "\r\n".join(self._buffer)
        del self._buffer[:]
>       self.send(msg)

[/usr/lib/python2.5/httplib.py:732]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

```
    def send(self, str):
        """Send `str' to the server."""
        if self.sock is None:
            if self.auto_open:
>               self.connect()

[/usr/lib/python2.5/httplib.py:699]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

    def connect(self):
        "Connect to a host on a given (SSL) port."

        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
>       sock.connect((self.host, self.port))

[/usr/lib/python2.5/httplib.py:1134]
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

E   [failure to get at sourcelines from <TracebackEntry /root/<string>:1>]
>   timeout: timed out

[/root/<string>:1]
- - - - - - - - - - - - - - - - - - - - - - - - -  test_create_eleve: recorded stdout - - - - - -
- - - - - - - - - - - - - - - - - - - -
Pas assez d'identifiants disponibles

___________________________________________________________________________________________________
_________________________________________
```

Demandes liées:

Lié à scribe-backend - Evolution #739: ajouter "entlogin:False" dans la créat...	**Ne sera pas résolu**	**30/06/2010**
Lié à zephir-parc - Anomalie #1772: codes ent : incohérence dans les plages r...	**Ne sera pas résolu**	**13/05/2011**

Révisions associées

Révision d917428a - 17/03/2006 18:32 - exarkun

Merge reliable-message-delivery-344

Author: exarkun
Reviewer: moe
Fixes #344, #413, #513, #514, #700, #705, #738; Refs #735

- Nit's command line is extended to accept a port number on which to run its webserver.

- The transport layer of Athena is replaced with a message-queue based mechanism. This supports gap and duplicate message detection, allowing the server to leave the page "connected" for a short while even if a client abruptly closes its connection(s). This should provide greater stability in the face of poor network conditions and various kinds of proxy misbehavior.

- All messages sent from the server while synchronously handling a message from the client are bundled up and sent in a single response, rather than putting each message into its own response. Likewise, messages sent from the client while synchronously handling a message from the server or synchronously handling a DOM event using <athena:handler> will also be bundled up in one request.

- Only one transport resource instance is created per LivePage, instead of one per request.

- The guaranteed namespace-aware getAttribute helper has been moved into the runtime support code.

- Several problems with error reporting on IE in the test suite have been fixed.

- A lingering usage of MochiKit Deferreds has been removed.
- Escape is now captured and handled with an explicit close message.
- If a page is unloaded because the user navigated to another page, the disconnect dialog is no longer displayed.
- The client-side debug log displays slightly more nicely.
- Radical once again avoids using active channels and constructs cacheable URLs for javascript modules.

Historique

#1 - 04/12/2012 10:58 - Joël Cuissinat

- Statut changé de Nouveau à Ne sera pas résolu

- Distribution mis à EOLE 2.3

Cette fonctionnalité n'est plus à la mode :)