**#17691** - https://redmine.vnc.biz/issues/17691

1. **Description**:

|  |
| --- |
| * 4 users are in a videoconference and they decide to use etherpad * they click on the button and then the etherpad session starts and they can work together with the document * **Problem:** * *the normal user does not know the etherpadID/padName (vnc.biz/etherpad/p/$ETHERPAD\_PAD\_ID)* * *Therefore the users are not able to get back to the etherpad they worked on after the videoconference is closed!* |

1. **Analytic**
   1. **Solutions**:

|  |
| --- |
| - When user click on Etherpad button, I send a message with etherpad-id to all participants.  - Handle message from jappix-mini and send email to all participants by using zimbra API.  - Use global variable MINI\_DICT\_CURRENT\_USER to check which user has been sent email  - Remove roomId in MINI\_DICT\_CURRENT\_USER when dispose video conference |

1. **Implementation**
   1. **Code:**

Edit etherpad.js

|  |
| --- |
| my.toggleEtherpad = function (isPresentation) {  if (!etherpadIFrame){  createIFrame();  //moderator send message start document sharing  if (Moderator.isModerator()) {  connection.emuc.sendMessage(domain + etherpadName + options + "&userName=", "etherpadCreated");  connection.emuc.sendMessage(domain + etherpadName + options + "&userName=");  }  }  ....... }; |

Edit muc.js

|  |
| --- |
| onMessage: function (msg) {  .......  if(nick == "etherpadCreated") {  //message for send email to all paticipants  return true;  }  ....... }; |

Edit mini.js

|  |
| --- |
| self.sendMail = function(list\_user, body\_text, subject\_text) {  // send email to list user  for (var i = 0; i < list\_user.length; i++) {  var body\_user = body\_text + list\_user[i].split('@')[0];  var soapDoc = AjxSoapDoc.create("SendMsgRequest", "urn:zimbraMail");  var m = soapDoc.set("m");  m.setAttribute("idnt", con.jid);  var node\_e = soapDoc.set("e", null, m);   node\_e.setAttribute("p", con.jid.split('@')[0]);  node\_e.setAttribute("a", con.jid);  node\_e.setAttribute("t", "f");   node\_e.setAttribute("p", list\_user[i].split('@')[0]);  node\_e.setAttribute("a", list\_user[i]);  node\_e.setAttribute("t", "t");  var node\_s = soapDoc.set("su", subject\_text, m);  var node = soapDoc.set("mp", null, m);  node.setAttribute("ct", "text/plain");  var subnode = soapDoc.set("content", body\_user, node);  var params = {  soapDoc: soapDoc,  asyncMode: true,  callback: (new AjxCallback(this, self.\_handleSOAPResponseXML)),  errorCallback: (new AjxCallback(this, self.\_handleSOAPErrorResponseXML))  };  appCtxt.getAppController().sendRequest(params);  }  };  ………………………..  self.handleMessage = function(msg) {  ......  if(body) {  if(jQuery(node).find('>nick[xmlns="http://jabber.org/protocol/nick"]').text() == "etherpadCreated" && self.getIndexObject(xid) == -1) {  var nick\_name = from.split("/")[1].replace("\_video","");  var current\_name = con.jid.split('@')[0];  var date\_time = new Date();  date\_time = date\_time.getFormatDate();   if(nick\_name == current\_name) {  // send email  var subject\_text = biz\_zcs\_vnc\_talk\_zimlet.vnc\_talk\_etherpad\_subject + ' ' + xid.split('@')[0] + ' - ' + date\_time;  var list\_all\_user = self.getAllParticipants(xid);  var list\_user = [];  for (var i = 0; i < list\_all\_user.length; i++) {  var to\_user = list\_all\_user[i].split('/')[0];  if(list\_user.indexOf(to\_user) == -1 && to\_user.indexOf('focus') == -1) {  list\_user.push(to\_user);  }  }   var email\_content = biz\_zcs\_vnc\_talk\_zimlet.vnc\_talk\_etherpad\_msg;  email\_content = email\_content.replace('ROOM\_NAME', xid.split('@')[0]);  email\_content = email\_content.replace('ETHERPAD\_DATE', date\_time);  email\_content = email\_content + ':\n' + body;   if(list\_user.length > 0) {  MINI\_DICT\_CURRENT\_USER.push(  {  key: xid,   value: {  "users": list\_user,  "subject": subject\_text,  "body": email\_content  }  });  self.sendMail(list\_user, email\_content, subject\_text);  }  }  return;  }  }  ...... };  ………………….  self.handlePresence = function(pr) {  ......  if (MINI\_DICT\_CURRENT\_USER.length > 0 && jid.indexOf('focus') == -1 && from.indexOf('\_video') != -1) {  for (var i = 0; i < MINI\_DICT\_CURRENT\_USER.length; i++) {  if (MINI\_DICT\_CURRENT\_USER[i].key == roomId) {  var msg\_value = MINI\_DICT\_CURRENT\_USER[i].value;  var jid\_value = jid.indexOf('/') == -1 ? jid : jid.split('/')[0];   if (msg\_value.users.indexOf(jid\_value) == -1) {  MINI\_DICT\_CURRENT\_USER[i].value.users.push(jid\_value);  // send email  self.sendMail([jid\_value], msg\_value.body, msg\_value.subject);  break;  }  }  }  }  ...... };  ……………………..  self.removeRoomFromCurrentUserDict = function (roomId) {  var index = self.getIndexObject(roomId);   if(index > -1) {  var participants = ROOMDETAIL.chatrooms[roomId];  var count = 0;  $.each(participants, function(index, value) {  if(value.fulljid.indexOf('focus') == -1 && value.fulljid.indexOf('VNCChat') == -1){  count++;  // break;  return;  }  });   if(count == 0) {  MINI\_DICT\_CURRENT\_USER.splice(index, 1);  }  }  }; |