**#21335** - <https://redmine.vnc.biz/issues/21335>

1. **Description**:

The workflow of checking a valid username should work for internal and external users.

Any entered username must be valid, depending on its characteristics.

This applies for the workflow of "Invite People" AND "Add People" buttons. Both methods/buttons should work for internal ANDexternal users. This is due usability issues to give the user always the same behavior!

The user may enter this valid input usernames formats:

* Internal/Normal username: (assuming vnc.biz as internal domain)
* rainer.schuth
* rainer.schuth@vnc.biz
* External usernames:
* phuong.vu@gmail.com

Invalid username examples:

* rainer\*schuth
* rainer\*schuth@vnc.biz
* rainer\*schuthvnc.biz@
* "rainer.schuth@vnc.biz"
* (rainer.schuth@gmail.com ; rainer.schuth@vnc.biz)
* rainer&schuth@gmail.com & rainer.schuth@vnc.biz

The $user.name should only contain valid characters. Then the string may contain only one valid [at] symbol but if and only if there is a valid domain after the [at]. Then the domain part(s) must also only contain valid characters.

If the invite username-string includes >= 1 special characters like "@#$%^&\*{}[]()" then display the error message AND abort creation of the room, invitation of user will also be aborted.

If user isn't in user-list, still send invitation to this users. It includes the case: someone invite external user(phuong.vu@vnc.biz) also display the error message.

To avoid multiple error prone checking you can always check if the entered usernames are valid emails. For example, if alice entered "rainer.schuth" then the code will append "@vnc.biz" to the "rainer.schuth" string creating the new username "rainer.schuth@vnc.biz". But only if the username exists in the COS list (contact list).

Then check iterate over the username list an check for invalid characters or invalid username structure.

For each valid username put it in the internal user list or the external user list.

For each invalid username put it in the invalid username array. Then abort the for-loop.

This way you have one or two for-loops which will efficiently iterate over all entered string looking for matching patterns.

After leaving the for-loop check invalid-user array if it has length > 0 then display the error message AND abort creation of the room, invitation user also.

Example:

|  |
| --- |
| usernameInputList := [ "rainer.schuth", "phuong.vu@vnc.biz", "quy.dang@gmail.com"] internalUsernames := [] externalUsernames := [] invalidUsernames := [] for $\_u in $usernameInputList:  $username := $\_u  if not isValidEmail($username): //Check if the $username contains maximal one [at]  $username := $username + $localEmailDomain //(e.g. Result= rainer.schuth@vnc.biz)  if isValidUsername($username):   //Check if the string has a valid format. $username@$tld.domain  //The username should not contain any invalid characters from "@#$%^&\*{}[]()". [at] is only allowed if it fits the email scheme.  if isInternalUser($username): //Check if the username is in the contact list  internalUsernames.add($username)  else:  externalUsernames.add($username)  else:  invalidUsernames.add($username)  break  if invalidUsernames.length() > 0:  abortRoomCreationInvitationProcedure() else:  createRoom()  inviteInteralUsers($internalUsernames)  inviteExternalUsers($externalUsernames) |

1. **Analytic**
   1. **Root cause**:
   2. **Solutions**:
2. **Implementation**
   1. **Code:**

* Add \_validateInvitees(usernameInputList) function:
* check valid emails.
* Check if the string has a valid format, username should not contain any invalid characters from "@#$%^&\*{}[]()".
* Return invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};

|  |
| --- |
| biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.\_validateInvitees = function(usernameInputList){  //Check if invalid username id is entered  var inviteeRegex = RegExp(/[`~@!#$%\^&\*+=\[\]\\';,/{}|\\":<>\?\\\)\(]/g);  var emailRegex = /^([\w-]+(?:\.[\w-]+)\*)@((?:[\w-]+\.)\*\w[\w-]{0,66})\.([a-z]{2,6}(?:\.[a-z]{2})?)$/i;  var invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};  var userList = usernameInputList.split(';');  //Pushing all invitees list into array and validating them.  for(var x = 0;x<userList.length;x++){  var currentUser = userList[x].trim();  if(currentUser.indexOf('@') != -1){  var isValidUser = emailRegex.test(currentUser);  if(isValidUser == false){  invitees.inValidUser.push(currentUser)  }else{  if(currentUser.split('@')[1] == MINI\_DOMAIN){  invitees.internal\_invitees.push(currentUser);  }else{  invitees.external\_invitees.push(currentUser)  }  }  }else{  if (inviteeRegex.test(currentUser)){  invitees.inValidUser.push(currentUser);  }else if ("" != currentUser){  invitees.internal\_invitees.push(currentUser + '@' + MINI\_DOMAIN);  }  }  }  return invitees; } |

* Modified \_invitePeopleDialogOkListener() function:
* Get list invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};
* check if invitees.inValidUser.length > 0, display the error message AND invitation of user will also be aborted.
* check if invitees.external\_invitees.length > 0, display the error message AND still invitation of internal-user.

|  |
| --- |
| biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.prototype.\_invitePeopleDialogOkListener = function() { ...  this.invitePeopleDialog.popdown();  var invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};  invitees = biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.\_validateInvitees(usernameInputList);   if(invitees.inValidUser.length > 0){  console.log('inValidUser\n', invitees.inValidUser);  var content = "Following are invalid username.Please correct it.<br/>";  for(var i=0; i<invitees.inValidUser.length; i++){  content = content + invitees.inValidUser[i]+"<br/>";  }  var msg = appCtxt.getMsgDialog();  msg.setMessage(content,DwtMessageDialog.WARNING\_STYLE);  msg.popup();  return;  }  if (invitees.external\_invitees.length > 0) {  var content = "External users can not be invited to a text chat conference.Please correct it.<br/>";  for(var i=0; i<invitees.external\_invitees.length; i++){  content = content + invitees.external\_invitees[i]+"<br/>";  }  var msg = appCtxt.getMsgDialog();  msg.setMessage(content,DwtMessageDialog.WARNING\_STYLE);  msg.popup();  };  var invitationList = invitees.internal\_invitees; ... } |

* Modified \_addPeopleDialogOkListener function:
* Get list invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};
* check if invitees.inValidUser.length > 0, display the error message AND abort creation of the room, invitation of user will also be aborted.

|  |
| --- |
| biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.prototype.\_addPeopleDialogOkListener = function() { ...  var inviteesResult = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};  inviteesResult = biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.\_validateInvitees(usernameInputList);  var invitees = {"internal\_invitees":[],"external\_invitees":[]};  invitees.internal\_invitees = inviteesResult.internal\_invitees;  invitees.external\_invitees = inviteesResult.external\_invitees;   ...  if(inviteesResult.inValidUser.length > 0){  var content = "Following are invalid username.Please correct it.<br/>";  for(var x=0;x<inviteesResult.inValidUser.length;x++){  content = content + inviteesResult.inValidUser[x]+"<br/>";  }  var msg = appCtxt.getMsgDialog();  msg.setMessage(content,DwtMessageDialog.WARNING\_STYLE);  msg.popup();  return;  } ... } |

* Prosody packaging, modified self.createGroupchatPrompt() function:
* Get list invitees = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};
* check if invitees.inValidUser.length > 0, display the error message AND invitation of user will also be aborted.
* check if invitees.external\_invitees.length > 0, display the error message AND still create room, invitation of internal-user.

|  |
| --- |
| self.createGroupchatPrompt = function() { ...  if ("" != invitees) {  var inviteesResult = {"internal\_invitees":[],"external\_invitees":[],"inValidUser":[]};  inviteesResult = biz\_zcs\_vnc\_talk\_zimlet\_HandlerObject.\_validateInvitees(invitees);  if (inviteesResult.inValidUser.length > 0) {  self.closePrompt();  var content = "Following are invalid username.Please correct it.<br/>";  for(var i=0; i<inviteesResult.inValidUser.length; i++){  content = content + inviteesResult.inValidUser[i]+"<br/>";  }  var msg = appCtxt.getMsgDialog();  msg.setMessage(content,DwtMessageDialog.WARNING\_STYLE);  msg.popup();  return;  };  if (inviteesResult.external\_invitees.length > 0) {  var content = "External users can not be invited to a text chat conference.Please correct it.<br/>";  for(var i=0; i<inviteesResult.external\_invitees.length; i++){  content = content + inviteesResult.external\_invitees[i]+"<br/>";  }  var msg = appCtxt.getMsgDialog();  msg.setMessage(content,DwtMessageDialog.WARNING\_STYLE);  msg.popup();  };  self.invitePeople = inviteesResult.internal\_invitees;  } else {  self.invitePeople = [];  } ... } |