Contents

[userExist 2](#_Toc349649963)

[groupExist 2](#_Toc349649964)

[getUserAttribute 2](#_Toc349649965)

[checkUserAttribute 2](#_Toc349649966)

[checkUserPassword 3](#_Toc349649967)

[checkGroupNameAndID 3](#_Toc349649968)

[checkUserSecondaryGroupMembership 3](#_Toc349649969)

[checkUserPrimaryGroupMembership 3](#_Toc349649970)

[checkUserLocked 3](#_Toc349649971)

[checkUserHasNoShellAccess 4](#_Toc349649972)

[checkUserGroupMembership 4](#_Toc349649973)

[checkUserChageAttribute 4](#_Toc349649974)

[setupUser 4](#_Toc349649975)

[setupGroup 5](#_Toc349649976)

[delGroup 5](#_Toc349649977)

[delUser 5](#_Toc349649978)

[checkUserFilePermission 5](#_Toc349649979)

[checkUserFileSpecialPermission 6](#_Toc349649980)

[checkGroupFilePermission 6](#_Toc349649981)

[checkOtherFilePermission 6](#_Toc349649982)

[checkNewlyCreatedFilesAttributes 7](#_Toc349649983)

UserGroup module

# userExist

userExist("john");

Checks if user exist.   
The return value is 0, if the user exist, and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |

# groupExist

groupExist("group1");

Checks if group exist. The return value is 0, if the user exist, and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Groupname (not GID!) | | group1 |

# getUserAttribute

getUserAttribute("john",”UID”);

It will return the wished attribute of the user.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | The wished attribute. | | UID - User ID  GID - Primary Group ID  DESC - Description of the user  HOME - Home of the user  SHELL - Shell of the user |

# checkUserAttribute

checkUserAttribute("john",”SHELL”,”/bin/bash”);

You can check the value of the selected attribute of the user.

The return value is 0, if the two values are equal, and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | The wished attribute. | | UID - User ID  GID - Primary Group ID  DESC - Description of the user  HOME - Home of the user  SHELL - Shell of the user |
| 3. | | Value of the attribute, you want ot compare with. | |  |

# checkUserPassword

checkUserPassword("john","secretPassword");

You can check the value of the selected attribute of the user.

The return value is 0, if the Password parameter equals with the password of the user, and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | Password | |  |

# checkGroupNameAndID

checkGroupNameAndID ("group1","556");

Checks the groups group ID (GID).

Return value is 0 if the GID related to the groupname

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Groupname | | group1 |
| 2. | | Group ID (GID) | | 556 |

# checkUserSecondaryGroupMembership

checkUserSecondaryGroupMembership("john","group2");

Checks if User is member of the group. It checks only the secondary groups.

Return value is 0 if the user is in group and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | Groupname (not GID!) | | group1 |

# checkUserPrimaryGroupMembership

checkUserPrimaryGroupMembership("john","group2");

Checks the users primary group.

Return value is 0 if the users primary group is correct and 1otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | Groupname (not GID!) | | group1 |

# checkUserLocked

checkUserLocked("john");

Checks if user is locked.   
The return value is 0 if it is locked and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |

# checkUserHasNoShellAccess

checkUserHasNoShellAccess("john");

Checks if user has no shell access.   
The return value is 0 if it the user has no shell access and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |

# checkUserGroupMembership

checkUserGroupMembership("john","groupX");

Checks if user is member of the given group. It checks both the primary and secondary groups.

Return value is 0 if the user is member of the group and 1 otherwise.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| parameter | descritpion | | example/values | | |
| 1. | | Username (not UID!) | | root |
| 2. | | Groupname (not GID!) | | group1 |

# checkUserChageAttribute

checkUserChageAttribute("john","EXPIRE\_DATE","2025-12-10");

You can check the value of the selected attribute of the user.

*Examples:*

The User tom's account should expire in 2025-12-10:

checkUserChageAttribute("tom","EXPIRE\_DATE","2025-12-10");

The warning days should be 12 before the password expires:

checkUserChageAttribute("tom","WARN\_DAYS","12");

The return Value is 0 when the attribute is correct, else it returns with 1.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Username (not UID!) | root |
| 2. | The wished attribute. | EXPIRE\_DATE - Account expires (date format: YYYY-MM-DD)  INACTIVE - Password inactive  MIN\_DAYS - Minimum number of days between password change  MAX\_DAYS - Maximum number of days between password change  WARN\_DAYS - Number of days of warning before password expires |
| 3. | The value you wish to compare with |  |

# setupUser

setupUser("john","4999","group1","group2,group3","/home/john","Descrition of user john","/bin/bash","true");

Creates user with the given settings. If you don’t declare any attribute, then it will be a default value.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Username (not UID!) | john |
| 2. | User ID | 4999 |
| 3. | Primary group | group1 |
| 4. | Secondary groups | group2,group3 |
| 5. | Home directory | /home/john |
| 6. | Description | Description of user john |
| 7. | Default shell | /home/john |
| 8. | Generate SSH key | true |

# setupGroup

setupGroup("group1","5678","john,mary");

Creates a group with the given Group ID and group members. If you don’t declare GID, then it will be set automatically.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Group name (not GID!) | group1 |
| 2. | Group ID (GID) | 5678 |
| 3. | Members | john,mary |

# delGroup

delGroup("group1");

Delete the group.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Group name (not GID!) | group1 |

# delUser

delUser("john");

Delete the user.

Delete the group.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Username (not UID!) | john |
| 2. (optional) | Delete the users home directory too | true |

# checkUserFilePermission

checkUserFilePermission(”john”,”/etc/passwd”,”r\*\*”);

Checks the users permissions on the given file.

The return value is 0 if the user has the given permissions.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Username (not UID!) | root |
| 2. | Filename with full path | /etc/passwd |
| 3. | Permissions, you want to check in UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |

# checkUserFileSpecialPermission

checkUserFileSpecialPermission(”/etc/passwd”,”SETUID”);

Checks the files special permissions.

The return value is 0 if the file has the given permissions.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Filename with full path | /etc/passwd |
| 2. | Permission you want to check It can be: SETUID, NO\_SETUID, SETGID, NO\_SETGID, STICKY, NO\_STICKY | SETUID |

# checkGroupFilePermission

checkGroupFilePermission(”group1”,”/etc/passwd”,”r\*\*”);

Checks the users permissions on the given file.

The return value is 0 if the user has the given permissions.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Groupname (not GID!) | group1 |
| 2. | Filename with full path | /etc/passwd |
| 3. | Permissions, you want to check in UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |

# checkOtherFilePermission

checkOtherFilePermission(”group1”,”/etc/passwd”,”r\*\*”);

Checks the others (not owner and owner group) permissions on the given file.

The return value is 0 if the „others” have the given permissions.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Filename with full path | /etc/passwd |
| 2. | Permissions, you want to check in UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |

# checkNewlyCreatedFilesAttributes

checkNewlyCreatedFilesAttributes(”/tmp/admins”,”group1”,”john”,”r\*\*”,”group2”,”rw-”,”---”);

It creates a new file in the given directory, and you can check, the owner group and the permissions of it. It returns 0, if every attribute is true and 1 otherwise.

|  |  |  |  |
| --- | --- | --- | --- |
| parameter | descritpion | example/values | |
| 1. | Directory | /tmp/admins |
| 2. | Owner Group (not GID!) | group1 |
| 3. | Groupname (not GID!) | group2 |
| 4. | Permissions of the group you set as 3rd parameter  In UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |
| 5. | USername (not UID!) | mary |
| 6. | Permissions of the user you set as 5th parameter  In UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |
| 7. | Permissions of Others (not owner and not member of the owner group)  In UGO format (rwx).  1.letter for read: r/-/\*  2.letter for write: w/-/\* 3.letter for exectue: x/-/\*  Use \* if you don't want to check, or it doesn't matter.  Use – if you want to deny it  e.g. r\*- means, that it has to be readable, it can be writable or not writable either, it musn't be executable | r\*\* |