Hvrvalidpw

Contents

Hvrvalidpw allows customization of how the HVR executable validates the username/password of incoming connections. This overrides the default behavior, which is to validate username/password as operating system credentials. **hvrvalidpw** is not a command to be executed manually in the command line to authenticate a user; it is only a plugin which is invoked by HVR for authentication. For more information about authentication modes and access control in HVR, see Authentication and Access Control.

HVR distribution includes the following authentication plugins:

- LDAP Authentication - Hvrvalidpwldap Plugin
- Private Password File Authentication - Hvrvalidpwfile Plugin
- Custom Hvrvalidpw Authentication

For HVR to invoke either of the above mentioned authentication plugins, the respective plugin file should be copied as **hvrvalidpw** in **HVR_HOME/lib/** directory.

LDAP Authentication - Hvrvalidpwldap Plugin

HVR authenticates the incoming username/password by invoking its **hvrvalidpwldap** plugin.This plugin authenticates a user by validating the credentials stored on LDAP server. This authentication is achieved by using the command file **hvrvalidpwldap** available in **HVR_HOME/lib** directory.
This plugin connects to the LDAP server with a search username and password. For Active Directory, it can connect using NTLM authentication. The search connection should have privileges to perform search operations. After establishing a connection with search user, an LDAP search is performed to validate the HVR user. User groups of the validated user also can be fetched from the LDAP server; these groups can be used inside the access control file.

hvrvalidpwldap is not a command to be executed manually in the command line to authenticate a user; it is only a plugin which is invoked by HVR for LDAP based authentication.

Installing Python Environment

HVR requires the LDAP python client module installed for using the LDAP authentication. Perform the following on HVR hub machine:

1. Install Python (only 2.7.x version is supported). Skip this step if the mentioned python version is already installed in the machine.
2. Install the following python client module:

```
pip install ldap3
```

Enabling LDAP Authentication

To enable LDAP authentication:

1. Create file **HVR_HOME/lib/hvrvalidpwldap.conf** to supply the configuration required for connecting to the LDAP server. The configuration file parameters are described in section LDAP Configuration File. An example configuration file **hvrvalidpwldap.conf_example** is available in **HVR _HOME/lib** directory.

2. HVR should use the username/password only for authentication, but must not change from the current operating system user to that login. To achieve this;

    - In Linux or Unix,

systemd

a. Set **user=** with a non-root operating system user.
b. Update the **ExecStart** from **-r** to **-r -A** to prevent changing of user.

xinetd

a. Set **user=** with a non-root operating system user.
b. Update the **server_args** from **-r** to **-r -A** to prevent changing of user.

inetd

a. Change the user from **root** to a non-root operating system user.
b. Update **-r** in the command as **-r -A** to prevent changing of user.

hvrremotelistener

a. Execute **hvrremotelistener** with option **-A** along with **-d** or **-i** options.

- In Windows,

  a. Execute **hvrremotelistener** with option **-A** along with **-ac** option in the command line. Option **-P** can also be used along with this command to create the service as non administrator operating system user.

     Option **-A** in **hvrremotelistener** is available only in CLI (an equivalent GUI option is not available in **Create Windows Service** dialog).

     Also note that using option **-ac** without option **-P** may require administrator privileges. In this case, it is recommended to use **Run as administrator** option while opening the Windows command line terminal.

3. Copy **HVR_HOME/lib/hvrvalidpwldap** to **HVR_HOME/lib/hvrvalidpw**.

   HVR only uses a plugin-based authentication system if it detects file **hvrvalidpw** in directory **HVR_HOME/lib**. This step activates **hvrvalidpwldap** plugin for user authentication.

LDAP Configuration File

This section lists and describes the parameters required for configuring the connection to the LDAP server in **hvrvalidpwldap.conf**.

Parameter	Description
LDAP_Server	The hostname or address of the LDAP server. Possible values are: • *hostname* • **ldap://***hostname* • **ldap://***hostname***:** *port* • **ldaps://***hostname* • **ldaps://***hostname***:** *port*

LDAP_Search_User	The credential to perform LDAP search. Possible values are: • **anonymous** : Do not bind to an LDAP user, perform search anonymously. • **self** : Bind to input HVR username/password. LDAP search will still be performed if the **LDAP_User_Method** is not set as **none**. • **self_ntlm** : Bind to input HVR username/password using the Active Directory NTLM authentication. LDAP search will still be performed if the **LDAP_User_Method** is not set as **none**. **Since** v5.6.5/1 • **user:***username* : Bind to LDAP user or distinguished name (DN) *username*. Requires **LDAP_Search_Password**. • **ntlm:***username* : Authenticate using the Active Directory NTLM. The format for *username* is *domain\user*. Requires **LDAP_Search_Password**.
LDAP_Search_Password	The password of the **LDAP_Search_User**.
LDAP_User_Method	The method to find (search) LDAP users. Possible values are: • **none** : To disable searching for the users from LDAP server. This can be used only if **LDAP_Search_User** is **self** or **self_ntlm**. Note that user groups cannot be searched in this case, so **LDAP_Group_Method** must also be set as **none**. **Since** v5.6.5/1 • *search_type***/***base_dn***/***filter* : A separate LDAP search is performed. Here, *search_type* is either **search_one** or **search_subtree**; *base_dn* is the starting point of search in LDAP tree; *filter* is an LDAP filter. Separator **/** can be replaced with any other non-alphanumeric character. In the example below, **%u** is replaced with the actual username.
LDAP_Group_Method	The method to find (search) LDAP user groups. Possible values are: • **none** : To disable searching for the user groups from LDAP server. • **user_attribute/***attr_name* : *attr_name* is an attribute of previously found user. • *search_type***/***base_dn***/***filter***/***attr_name* : A separate LDAP search is performed. Here, *search_type* is either **search_one** or **search_subtree**; *base_dn* is the starting point of search in LDAP tree; *filter* is an LDAP filter; *attr_name* is an attribute of found group entities to use as group name. Separator **/** can be replaced with any other non-alphanumeric character. In the example below, **%U** is replaced with found user's distinguished name (DN).
LDAP_Timeout	Timeout (in seconds) for the LDAP connections and queries.
LDAP_Error_Trace **Since** v5.6.5/1	Show/hide detailed information (to help diagnose configuration problems or error messages) in case of authentication failure. Possible values are: **0** : To disable error tracing. **1** : To enable error tracing. Enabling error tracing will allow unauthorized users to see details of authentication failure.

Examples

For basic Active Directory setup,

```
LDAP_Server=localhost

LDAP_Search_User=self_ntlm

LDAP_User_Method=none

LDAP_Group_Method=none

LDAP_Timeout=10
```

For basic LDAP setup,

```
LDAP_Server=localhost

LDAP_Search_User=self

LDAP_User_Method=none

LDAP_Group_Method=none

LDAP_Timeout=10
```

For basic setup with a dedicated lowest-privileged search user,

```
LDAP_Server=localhost

LDAP_Search_User=user:CN=SearchUser,CN=Users,DC=organization,DC=local

LDAP_Search_Password=password

LDAP_User_Method=search_subtree/CN=Users,DC=organization,DC=local/(&
(objectClass=person)(|(cn=%u)(sAMAccountName=%u)(uid=%u)))

LDAP_Group_Method=search_subtree/DC=organization,DC=local/(&
(objectClass=group)(member=%U))/CN

LDAP_Timeout=10
```

Specify the **LDAP_User_Method** and **LDAP_Group_Method** that is appropriate for your LDAP setup.

Files

HVR_HOME

lib

hvrvalidpwldap	The plugin file for LDAP authentication. To use authentication through LDAP this file should be copied to **hvrvalidpw**.
hvrvalidpwldap.conf	Configuration for this plugin.
hvrvalidpwldap.conf_example	Example configuration file for this plugin.
hvrvalidpw	Used by HVR for user authentication. For LDAP authentication, this should be a copy of **hvrvalidpwldap**.

Private Password File Authentication - Hvrvalidpwfile Plugin

HVR authenticates incoming username/password by invoking its **hvrvalidpwfile** plugin. This plugin authenticates a user by validating the credentials stored in a private password file. This authentication is achieved by using the command file **hvrvalidpwfile** available in **HVR_HOME/lib** directory.
For authentication, this plugin is invoked by HVR without any arguments and supplies the login and password (space separated) on the standard input.

Enabling Private Password File Authentication

To enable Private Password File Authentication:

1. Create user:
    a. Execute command **hvrvalidpwfile** *username*. For example,

    ```
    perl hvrvalidpwfile user1
    ```

    b. Enter password at the prompt and press **Enter** key.
    c. Repeat steps a and b to create multiple users.

    The username and password are stored in **HVR_HOME/lib/hvrpasswd**. For more information, see Managing Usernames and Passwords.
2. HVR should use the username/password only for authentication, but must not change from the current operating system user to that login. To achieve this;
    - In Linux or Unix,

      systemd

      a. Set **user=** with a non-root operating system user.
      b. Update the **ExecStart** from **-r** to **-r -A** to prevent changing of user.

      xinetd

      a. Set **user=** with a non-root operating system user.
      b. Update the **server_args** from **-r** to **-r -A** to prevent changing of user.

      inetd

      a. Change the user from **root** to a non-root operating system user.
      b. Update **-r** in the command as **-r -A** to prevent changing of user.

      hvrremotelistener

      a. Execute **hvrremotelistener** with option **-A** along with **-d** or **-i** options.
    - In Windows,

      a. Execute **hvrremotelistener** with option **-A** along with **-ac** option in the command line. Option **-P** can also be used along with this command to create the service as non administrator operating system user.
3. Copy **HVR_HOME/lib/hvrvalidpwfile** to **HVR_HOME/lib/hvrvalidpw**.

    HVR only uses a plugin-based authentication system if it detects file **hvrvalidpw** in directory **HVR_HOME/lib** . This step activates **hvrvalidpwfile** plugin for user authentication.

Managing Usernames and Passwords

The command **hvrvalidpwfile** allows you to manage usernames and passwords. The password is always encrypted and stored in the custom password file **hvrpasswd** available in **HVR_HOME/lib** directory.

- To create new user or update the password of an existing user:
  hvrvalidpwfile *username*

  This command prompts to enter password. The password entered in this command is saved for the respective *username*.
- To create new user or update the password of an existing user without displaying prompt to enter password:
  hvrvalidpwfile -b *username password*

- To delete an existing user:
  hvrvalidpwfile -D *username*

Files

HVR_HOME

lib

hvrvalidpwfile	The plugin file for private password file authentication. This file should be copied to **hvrvalidpw**.
hvrpasswd	Used by **hvrvalidpwfile** for storing the username and password.
hvrvalidpw	Used by HVR for user authentication. For local password file authentication, this should be a copy of **hvrvalidpwfile**.

Custom Hvrvalidpw Authentication

HVR also allows you to supply your own **hvrvalidpw** authentication plugin. This plugin can be a modified version of **hvrvalidpwfile** plugin or else you can create your own plugin. The custom plugin file should be named **hvrvalidpw** and saved in **HVR_HOME/lib** directory. It should obey the following calling conventions:

- It should read a line of input which will contain the username and password.
- It should exit with code 0 if the username and password is valid. Otherwise, it should exit with code 1.