forcefsck_verteilen

```
#!/bin/bash
# set -x; # DEBUG OPTION - uncomment to activate



# Script um auf mehreren Hosts die Datei /forcefsck im root-Dateisystem zu erzeugen.
# ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------
#
#
#    /!\ ACHTUNG: Benoetigt HPU-Rechte zur Ausfuehrung /!\
#
#
# Vor Verwendung muss eine Datei angelegt werden, die eine Liste mit Hostnamen enthält (jeweils ein
Hostname pro Zeile), auf denen die Datei /forcefsck angelegt werden soll.
# Der vollstaendige Pfad (relativ oder absolut) inklusive Dateiname muss als erstes Argument uebergeben
werden.
#
# Ausserdem wird eine Datei benoetigt, die das LDAP-Passwort des ausfuehrenden Users enthaelt. Wenn der
volle Pfad zu dieser Datei nicht als zweites Argument uebergeben wird, dann wird
# per Default versucht auf die Datei ~/LDAPPASSWORD.TXT zuzugreifen.
#
# Es empfiehlt sich, das LDAP-Passwort in die Zwischenablage zu kopieren, damit man es zur
Authentifizierung fuer sudo einfach in die Shell pasten kann.
#
# Dieses Script dient der Vorbereitung von Changes.
#
# Anwendung:
# ----------
#
#            forcefsck_verteilen.sh /Pfad/zur/Liste/mit/Hostnamen.txt </Pfad/zur/Datei/mit/dem/LDAP-
Passwort.txt>
#
# ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------



# ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------
#
# VARIABLEN DEKLARATION/INITIALISIERUNG UND FUNKTIONEN
#
# ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------

    # ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------
    # Variablen
    # ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------

    # VAR_USER: den Usernamen des ausfuehrenden Users zuweisen
    VAR_USER="${USER}";

    # VAR_HOSTLISTENPFAD: Voller Pfad zur Datei (inklusive Dateiname), die eine Liste mit Hosts auf
denen die Datei /forcefsck angelegt werden soll enthaelt
    VAR_HOSTLISTENPFAD=$1;

    # VAR_LDAPPASSWORTPFAD: Initialisierung der Variable mit dem Wert aus dem zweiten uebergeben
Argument. Falls das zweite Argument unset oder null ist wird der Default-Wert übernommen.
    VAR_LDAPPASSWORTPFAD=${2-~/LDAPPASSWORD.TXT};

    # VAR_REDHATRELEASEPFAD: Pfad zur Datei /etc/redhat-release
    VAR_REDHATRELEASEPFAD="/etc/redhat-release";
```

```
    # VAR_TMPISTREDHAT: Initialisierung der Variable
    VAR_TMPISTREDHAT=false;

    # VAR_TMPRHELVERSION: Initialisierung der Variable
    VAR_TMPRHELVERSION="";

    # VAR_TMPCONERROR: Initialisierung der Variable
    VAR_TMPCONERROR="";

    # VAR_LISTEHOSTS: Initialisierung der Variable fuer die von nicht erreichbaren Systemen bereinigte
Liste an Hosts
    VAR_LISTEHOSTS="";

    # VAR_INITIALLISTEHOSTS: Initialisierung der Variable fuer die durch den User uebergebenen Liste an
Hosts
    VAR_INITIALLISTEHOSTS="";

    # VAR_LISTERHELHOSTS: Initialisierung der Variable fuer eine Liste mit RHEL Hosts
    VAR_LISTERHELHOSTS="";

    # VAR_LISTEANDEREHOSTS: Initialisierung der Variable fuer eine Liste von Hosts auf denen nicht RHEL
installiert ist oder die nicht RHEL Version 6/7 installiert haben
    VAR_LISTEANDEREHOSTS="";

    # VAR_LISTECONERROR: Initialisierung der Variable fuer eine Liste von Hosts auf die nicht per SSH
connected werden konnte
    VAR_LISTECONERROR="";

    # VAR_TEMP: Initialisierung der Variable
    VAR_TEMP="";



    # ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
    # Funktionen
    # ------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

    #-------------------------------------------------------------------------------
    # Funktion:     f_test_redhat()
    # Beschreibung: Prueft anhand der Existenz der Datei /etc/redhat-release ob
    #               das Betriebssystem ein redhat linux ist
    # Return code:  keiner - setzt den Wert der Variable VAR_TMPISTREDHAT (bool)
    #
    # Parameter:    1: remote hostname
    #-------------------------------------------------------------------------------
    f_test_redhat() {
        # Variablen
        local L_HOSTNAME="$1";                                # Der Variable L_HOSTNAME wird der Wert des
ersten Arguments zugewiesen
        local L_LDAPPASSWORTPFAD="$VAR_LDAPPASSWORTPFAD";    # Der Variable L_LDAPPASSWORTPFAD wird der
Wert aus der globalen Variable VAR_LDAPPASSWORTPFAD zugewiesen
        local L_REDHATRELEASEPFAD="$VAR_REDHATRELEASEPFAD";  # Der Variable L_REDHATRELEASEPFAD wird der
Wert aus der globalen Variable VAR_REDHATRELEASEPFAD zugewiesen
        local L_LASTSTATUS="";                                # Die Variable L_LASTSTATUS wird
initialisiert

        # Das LDAP-Passwort wird per sshpass an ssh uebergeben. Hostkey- und Host-IP-Checking werden
deaktiviert. Auf dem Remote-System
        # wird geprueft ob die Datei /etc/redhat-release existiert.
        sshpass -f $L_LDAPPASSWORTPFAD ssh -o StrictHostKeyChecking=no -o CheckHostIP=no $L_HOSTNAME
"test -e $L_REDHATRELEASEPFAD";

        # Der Rueckgabewert des Befehls test wird der Variable L_LASTSTATUS zugewiesen
        L_LASTSTATUS=$?;

        # Der globalen Variable VAR_TMPISTREDHAT wird je nach dem Rückgabewert in L_LASTSTATUS (0 -
erfolgreich, 1 - nicht erfolgreich) entweder true oder false zugewiesen
        case "$L_LASTSTATUS" in
```

```
        0) VAR_TMPISTREDHAT="true"
           ;;
        *) VAR_TMPISTREDHAT="false"
           ;;
        esac
    }



    #-----------------------------------------------------------------------------
    # Funktion:     f_get_redhat_major_version()
    # Beschreibung: Ermittelt aus der in /etc/redhat-release angegebenen
    #               Versionsnummer die Major-Versionsnummer
    # Return code:  keiner - setzt den Wert der Variable VAR_TMPRHELVERSION
    #
    # Parameter:    1: remote hostname
    #-----------------------------------------------------------------------------
    f_get_redhat_major_version() {
        # Variablen
        local L_HOSTNAME="$1";                                # Der Variable L_HOSTNAME wird der Wert des
ersten Arguments zugewiesen
        local L_LDAPPASSWORTPFAD="$VAR_LDAPPASSWORTPFAD";    # Der Variable L_LDAPPASSWORTPFAD wird der
Wert aus der globalen Variable VAR_LDAPPASSWORTPFAD zugewiesen
        local L_REDHATRELEASEPFAD="$VAR_REDHATRELEASEPFAD";  # Der Variable L_REDHATRELEASEPFAD wird der
Wert aus der globalen Variable VAR_REDHATRELEASEPFAD zugewiesen
        local L_REDHATVERSION="";                            # Die Variable L_REDHATVERSION wird
initialisiert
        local L_REDHATMAJORVERSION="";                       # Die Variable L_REDHATMAJORVERSION wird
initialisiert

        # Das LDAP-Passwort wird per sshpass an ssh uebergeben. Die Ausgabe von ssh wird unterdrückt.
Hostkey- und Host-IP-Checking werden deaktiviert.
        # Auf dem Remote-System wird der Inhalt der Datei /etc/redhat-release ausgegeben. Diese Ausgabe
wird der Variable L_REDHATVERSION zugewiesen.
        L_REDHATVERSION=$(sshpass -f $L_LDAPPASSWORTPFAD ssh -q -o StrictHostKeyChecking=no -o
CheckHostIP=no $L_HOSTNAME "cat $L_REDHATRELEASEPFAD");

        # Per grep wird in der Here-Variable L_REDHATVERSION nach einem Teilstring gesucht, der aus
Zahlen oder Punkten besteht. Durch die Option -o wird dieser ausgegeben und der Variable L_REDHATVERSION
zugewiesen.
        L_REDHATVERSION=$(grep -Eo '[0-9.]+' <<<$L_REDHATVERSION);

        # Per Parameter-Expansion wird aus dem Wert von L_REDHATVERSION der Teilstring vor dem "." als
Wert der Variable L_REDHATMAJORVERSION zugewiesen (macht z. B. "5" aus "5.11")
        # L_REDHATMAJORVERSION="${$L_REDHATVERSION%\..+}";
        L_REDHATMAJORVERSION=$( cut -d "." -f1 <<<"$L_REDHATVERSION" );

        # Der globalen Variable VAR_TMPRHELVERSION wird der Wert aus L_REDHATMAJORVERSION zugewiesen
        VAR_TMPRHELVERSION="$L_REDHATMAJORVERSION";
    }



    #-----------------------------------------------------------------------------
    # Funktion:     f_test_ssh_connection()
    # Beschreibung: testet ob eine SSH-verbindung zu dem uebergebenen Hostnamen
    #               aufgebaut werden kann
    # Return code:  keiner - setzt den Wert der Variable VAR_TMPCONERROR
    #
    # Parameter:    1: remote hostname
    #-----------------------------------------------------------------------------
    f_test_ssh_connection() {
        # Variablen
        local L_HOSTNAME="$1";                                # Der Variable L_HOSTNAME wird der Wert des
ersten Arguments zugewiesen
        local L_LDAPPASSWORTPFAD="$VAR_LDAPPASSWORTPFAD";    # Der Variable L_LDAPPASSWORTPFAD wird der
Wert aus der globalen Variable VAR_LDAPPASSWORTPFAD zugewiesen
        local L_LASTSTATUS="";                                # Die Variable L_LASTSTATUS wird
initialisiert
```

```
        # Das LDAP-Passwort wird per sshpass an ssh uebergeben. Die Ausgabe von ssh wird unterdrückt.
Hostkey- und Host-IP-Checking werden deaktiviert.
        # Auf dem Remote-System wird der Befehl "exit" ausgeführt.
        sshpass -f $L_LDAPPASSWORTPFAD ssh -q -o StrictHostKeyChecking=no -o CheckHostIP=no $L_HOSTNAME
"exit";

        # Der Rueckgabewert des Befehls "exit" wird der Variable L_LASTSTATUS zugewiesen
        L_LASTSTATUS=$?;

        # Der globalen Variable VAR_TMPCONERROR wird je nach dem Rückgabewert in L_LASTSTATUS (0 -
erfolgreich, > 0 - nicht erfolgreich) entweder true oder false zugewiesen
        case "$L_LASTSTATUS" in
        0) VAR_TMPCONERROR="false"
           ;;
        *) VAR_TMPCONERROR="true"
           ;;
        esac
    }



# ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
#
# PROGRAMMABLAUF
#
# ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
    # Pruefen ob alle Argumente uebergeben wurden und ob diese den Vorgaben entsprechen (Datei existiert
und ist nicht leer)
    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

    # Pruefen ob ueberhaupt Argumente uebergeben wurden und ob mehr als eines uebergeben wurde (es wird
mindestens ein Argument erwartet)
    if [ $# -eq 0 ]
    then
        # Es wurden keine Argumente uebergeben
        echo "ERROR: Es wurden keine Argumente uebergeben";
        exit 1; # Skript terminieren und Fehler signalisieren
    else
        # Es wurde mindestens ein Argument übergeben
        if [ $# -gt 2 ]
        then
            # Meldung ausgeben und abbrechen wenn mehr als zwei Argumente uebergeben wurde
            echo "ERROR: Es wurden mehr als zwei Argumente uebergeben";
            exit 1; # Skript terminieren und Fehler signalisieren
        else
            # Es wurde eine gueltige Anzahl Argumente uebergeben
            # Testen ob das Argument der Pfad zu einer Datei ist, die existiert und nicht 0 Bytes gross
ist
            if [ -s "$VAR_HOSTLISTENPFAD" ]
            then
                # Das Argument ist ein Pfad zu einer Datei die nicht leer ist
                echo "Uebernehme Hostliste aus: ${VAR_HOSTLISTENPFAD}";
            else
                echo "ERROR: Die Datei ${VAR_HOSTLISTENPFAD} kann nicht gefunden werden oder sie ist
leer.";
                exit 1; # Skript terminieren und Fehler signalisieren
            fi

            # Testen ob die in VAR_LDAPPASSWORTPFAD angegebene Datei (entweder übergeben oder per
Defaul-Value) existiert und nicht 0 Bytes gross ist
            if [ -s "$VAR_LDAPPASSWORTPFAD" ]
            then
                # Das Argument ist ein Pfad zu einer Datei die nicht leer ist
                echo "Uebernehme das LDAP-Passwort aus: ${VAR_LDAPPASSWORTPFAD}";
```

```
            else
                echo "ERROR: Die Datei ${VAR_LDAPPASSWORTPFAD} kann nicht gefunden werden oder sie ist
leer.";
                exit 1; # Skript terminieren und Fehler signalisieren
            fi
        fi
    fi



    # ------------------
```

```
        for ERRORHOST in $VAR_LISTECONERROR
        do
            echo "${ERRORHOST}";
        done

        # Skript abbrechen und Fehler signalisieren
        exit 1;
    fi



    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
    # Erstelle eine Liste mit Hosts, auf denen die Datei forcefsck benoetigt wird um den Filesystemcheck
auszuloesen, sowie eine Liste mit Hosts mit anderen Betriebssystemen
    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

    # Die Hostnamen in VAR_LISTEHOSTS werden in einer Schleife durchlaufen
    for TMPPOINTERA in $VAR_LISTEHOSTS
    do
        # Mit der Funktion f_test_redhat wird auf dem remote-Host geprueft ob die Datei /etc/redhat-
release existiert und der Variable VAR_TMPISTREDHAT entweder true oder false zugewiesen
        f_test_redhat "${TMPPOINTERA}";

        # Der Wert der Variable VAR_TMPISTREDHAT wird geprueft
        case "$VAR_TMPISTREDHAT" in

        false) # Der Wert ist "false" - Das System ist kein redhat linux
               # Ein Leerzeichen und der Hostname wird an den Wert in Variable VAR_LISTEANDEREHOSTS
angehaengt
               VAR_LISTEANDEREHOSTS="${VAR_LISTEANDEREHOSTS} ${TMPPOINTERA}";
               ;;

        *)     # Alle anderen Werte - Das System ist ein redhat linux
               # Mit der Funktion f_get_redhat_major_version wird die Major-Version des installierten
redhat linux auf dem Remote-System ermittelt
               # und der Variable VAR_TMPRHELVERSION zugewiesen
               f_get_redhat_major_version "${TMPPOINTERA}"

               # Der Wert in VAR_TMPRHELVERSION wird ueberprüft
               case "$VAR_TMPRHELVERSION" in

                   6|7) # Der Wert (die Major-Version) entspricht entweder "6" oder "7"
                        # Ein Leerzeichen und der Hostname wird an den Wert von VAR_LISTERHELHOSTS
angehaengt
                        VAR_LISTERHELHOSTS="${VAR_LISTERHELHOSTS} ${TMPPOINTERA}";
                        ;;

                   *)   # Die Major-Version ist nicht "6" oder "7"
                        # Ein Leerzeichen und der Hostname wird an den Wert von VAR_LISTEANDEREHOSTS
angehaengt
                        VAR_LISTEANDEREHOSTS="$VAR_LISTEANDEREHOSTS ${TMPPOINTERA}";
                        ;;
               esac
               ;;
        esac
    done;



    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
    # Anlegen der lokalen Datei ~/forcefsck um sie per scp auf die Hosts verteilen zu koennen
    # ---------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------

    # Die Datei forcefsck im User-Home auf login11/12 erzeugen, damit diese spaeter per scp auf die
Hosts verteilt werden kann.
    touch ~/forcefsck;
```

```
    # ------------------
```

```
"==============================================================================================
==============================";
    echo "Folgende Hosts sind keine RHEL 6/7-Systeme, die Datei /forcefsck wurde auf ihnen nicht
erzeugt:";
    echo
"==============================================================================================
==============================";
    echo "";

    for TMPPOINTERD in $VAR_LISTEANDEREHOSTS
    do
        echo "${TMPPOINTERD}";
    done;

exit 0;
```

From:
https://wiki.nanoscopic.de/ - **nanoscopic wiki**

Permanent link:
https://wiki.nanoscopic.de/doku.php/pages/scripts/forcefsck_verteilen

Last update: **2022/12/31 00:33**