SUSE Manager / Uyuni - Database Backup with "smdba" fails on symlink

Date:	2021-08-31
modified:	2021-08-31
tags:	SUSE Manager, Uyuni, Database, Backup
description:	smdba backup can't handle symlinks
category:	Linux
slug:	suse-manager-uyuni-database-backup-with-smdba-fails-on-symlink
Author:	Dominik Wombacher
lang:	en
transid:	suse-manager-uyuni-database-backup-with-smdba-fails-on-symlink
Status:	published

I performed a Upgrade of our SUSE Manager Instance at work to Version 4.2 and in parallel adjusted the disk layout a bit.

The exact details about all the changes are not that interesting, more the fact which issue I faced after changing */var/spacewalk/db-backup* to be a symlink into a sub-folder of a new mountpoint.

Backups are created and handled by the smdba tool, which performs an Owner and Permission check as soon a Backup was started. The path is defined by the argument "--backup-dir", default as mentioned in the official docs is */var/spacewalk/db-backup*, but as of today it doesn't follow symlinks.

So what happens: It will check the owner and permissions of the symlink instead the target folder. Due to the limitations that chmod can't be performed on symlinks, it stays 777, smdba will always fail when comparing with */var/lib/pgsql/data*.

To solve this issue and make smdba symlink aware, some small adjustments on the code of */usr/lib/python3.6/site-packages/smdba/postgresqlgate.py* are required.

I created a GitHub Issue (Archive: [1], [2]) to report my findings and discuss a solution.

Based on the feedback and suggestions from Victor and Michael, I created a Pull Request to get the fix hopefully included in a future release :)

```
diff --git a/src/smdba/postgresqlgate.py b/src/smdba/postgresqlgate.py
index 5bcb86e..e28b8f2 100644
--- a/src/smdba/postgresqlgate.py
+++ b/src/smdba/postgresqlgate.py
@@ -767,6 +767,11 @@ def do_backup_hot(self, *opts: str, **args: str) -> None:  # pylint: disable=W06
         if 'enable' in args.keys():
             # Check destination only in case user is enabling the backup
             if args.get('enable') == 'on':
+                # Save original value in temporary variable to re-assign after Permission Check
+                args_backup_dir_orig = args['backup-dir']
+                # If backup-dir is Symlink, use target instead
+                while os.path.islink(args['backup-dir']):
+                    args['backup-dir'] = os.readlink(args_backup_dir_orig)
                 # Same owner?
                 if os.lstat(args['backup-dir']).st_uid != os.lstat(self.config['pcnf_pg_data']).st_uid \
                         or os.lstat(args['backup-dir']).st_gid != os.lstat(self.config['pcnf_pg_data']).st_gid:
@@ -777,6 +782,8 @@ def do_backup_hot(self, *opts: str, **args: str) -> None:  # pylint: disable=W06
                 if oct(os.lstat(args['backup-dir']).st_mode & 0o777) != oct(os.lstat(self.config['pcnf_pg_data']).st_mode & 0o777):
                     raise GateException("The \"%s\" directory must have the same permissions as \"%s\" directory."
                                         % (args['backup-dir'], self.config['pcnf_pg_data']))
+                # Avoid issues at a later point due to different paths by setting original value
+                args['backup-dir'] = args_backup_dir_orig
             self._perform_enable_backups(**args)

         if 'source' in args.keys():
```

I know that I could also use a different path in the "--backup-dir" argument, which already points to the new location. But in my opinion it's helpful to stick with paths that are mentioned in the official documentation when there are multiple administrators.

Doesn't matter how good your internal KB is, in case of an issues or when new people takeover, project / vendor guides mostly have a higher precedence.

Also supporting Symlinks avoid that configured backup cronjobs silently failing in case someone has the idea to move the folder away and configure a symlink, like me ;)