DPM Use Cases

Package Lifecycle Management

Package lifecycle management includes:

1. Install a package.
2. Update a package.
3. Downgrade a package.
4. Remove a package.
5. Reinstall a package.

With the exception of removing a package, all operations require the package of the version being installed to be supplied to the utility. DPM does not cache packages.

When reinstalling, updating, or downgrading, all non-controlled files are optionally forcibly replaced if supplied the option to tell it to do so.

When removing a package, only controlled files are removed unless the option to include non-controlled files is supplied.

Installed Package Reporting

DPM can query it's `DPMDB` of installed packages for any of the attributes stored in the `DPM Backing Tree` to provide the user with information about any or all of the packages installed on the system.

1. list all packages installed
2. list which installed package(s) provide which file path on the system, such as a shared object or binary path
3. list which files an installed package provides by package name, including controlled and non-controlled files.
4. list installed packages by name whose properties match supplied expressions for AUTHOR, MAINTAINER, ARCHITECTURE,

DESCRIPTION, LICENSE, NAME, SOURCE, PROVIDES, REPLACES

5. check the file integrity of an installed package (controlled files only by default, but can optionally include non-controlled files if the option is provided by the user) using the `CONTENTS_MANIFEST_DIGEST` and `PACKAGE_DIGEST` checksums to compare against files deployed on the system by that package.
6. check the signature validity of the metadata for an installed package to ensure tampering has not occurred. this requires the package to be gpg signed and will fail if it is not signed.
7. check the file integrity for all installed packages on the system
8. check the signature validity for all installed packages on the system
9. list the ARCHITECTURE, AUTHOR, MAINTAINER, DESCRIPTION, LICENSE, NAME, PACKAGE_DIGEST, PROVIDES, REPLACES, SOURCE, VERSION of an installed package
10. list the immediate dependencies of an installed package
11. list the computed dependency tree of an installed package
12. list the hooks of an installed package by supplying the hook's reserved filename.
13. check the hooks' file integrity for an installed package
14. check the hooks' file integrity for all installed packages
15. list the cryptographic hashing method of a package
16. list all packages installed whose cryptographic hashing method matches a supplied pattern
17. list the full dependency tree map of all packages on the system
18. list the audit history of package changes