DBus Permission Manager

Концепция

В популярных мобильных ОС присутствует динамическое управление разрешениями. Один из способов его реализации - централизованный сервис, у которого стороннее приложение может запросить какое-либо разрешение, а другой сервис может узнать, запрашивало ли приложение, которое к нему обращается, определенный вид разрешения.

Необходимо

1. Реализовать DBus сервис с именем `com.system.permissions` на сессионной шине с использованием языка C++, который имеет 2 метода:
    - `void RequestPermission(permissionEnumCode: int)`
      Который получает путь до исполняемого файла, вызвавшего данный метод, по dbus имени клиента (пример, как это можно сделать) и сохраняет в базу данных `SQLite` информацию о том, что данный исполняемый файла запросил заданное разрешение. В случае ошибки метод должен возвращать DBus ошибку с человекочитаемым сообщением.

    - `bool CheckApplicationHasPermission(applicationExecPath: String, permissionEnumCode: int)`
      Который проверяет имеется ли у приложение с заданным путем до исполняемого файла заданное разрешение. В случае ошибки метод должен возвращать DBus ошибку с человекочитаемым сообщением.

      Перечисление разрешений:
      `enum Permissions { SystemTime = 0 }`

      Пример использования:

      ```
      gdbus send -e -d com.system.permissions -o / -m com.system.permissions.RequestPermission 0
      gdbus send -e -d com.system.permissions -o / -m com.system.permissions.CheckApplicationHasPermission
      /usr/bin/com.example.example 0
      ```

2. Реализовать DBus сервис `com.system.time` на сессионной шине с использованием языка C++, который имеет 1 метод:
    - `uint64 GetSystemTime()`
      Который возвращает `timestamp` текущего системного времени. Однако перед этим, он получает путь до исполняемого файла, вызвавшего данный метод, по dbus имени клиента (пример, как это можно сделать) и проверяет при помощи DBus сервиса `com.system.permissions`, имеет ли данный исполняемый файл разрешение `SystemTime`. В случае если исполняемый файл не имеет разрешения `SystemTime`, должна возвращаться ошибка `UnauthorizedAccess` с человекочитаемым сообщением. В случае любой другой ошибки должна возвращаться обычная ошибка с человекочитаемым сообщением.

      Пример использования:
      `gdbus send -e -d com.system.time -o / -m com.system.time.GetSystemTime`

3. Реализовать приложение с использованием языка C++, которое:
    - Пробует запросить у сервиса `com.system.time` текущее время и в случае ошибки `UnauthorizedAccess` запрашивает разрешение `SystemTime` у сервиса `com.system.permissions`, после чего пытается повторить запрос текущего времени.
    - После получения `timestamp`'а текущего времени, выводит его на экран в человекочитемом виде.

Обязательные требования

1. Проект должен быть написан на C++ с использованием библиотек `sdbus-c++` или `QtDBus`
2. Проект должен быть размещен на GitHub с подробным README с инструкцией по сборке и использованию
3. В проекте должна быть осмысленная история коммитов
4. Проект должен компилироваться под Linux (например Ubuntu 20.04/22.04)
5. Проект должен компилироваться без предупреждений компилятора
6. Проект должен использовать систему сборки CMake, Meson или QMake
7. Проект должен быть отформатирован при помощи clang-format