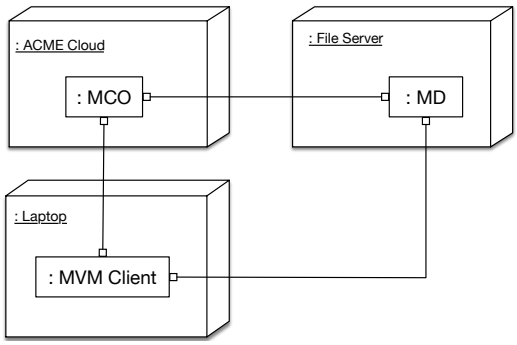
Exercise: Architecture

Background

ACME Water commissioned Midnight Engineering (their preferred system integrator) to implement their 'Midnight Version Manager' product based on ACME Water's version control requirements. MVM is an old, but well established product. It is already used in some ACME Water plants, but not by instrument technicians. ACME Water now intend rolling MVM out to all instrument technician laptops, and this will be the future means by which software changes will be version controlled.

About Midnight Version Manager

Midnight Version Manager (MVM) is a multi-user, distributed version of RCS: an open source file based revision control system. MVM consists of a GUI application, and two services which run on corporate servers. All software applications have been written in Visual C++, and run on Windows PCs, laptops, and server infrastructure.



The MVM software architecture is illustrated in the above UML deployment diagram, and consists of three main components:

MVM Client application (MVM Client)

This is a GUI application that runs on PCs or laptops. Users will interact with a *Telemetry* MVM client to make changes to telemetry software, and separate *PLC* and *SCADA* MVM clients when making changes to PLC or SCADA configuration files respectively.
When users 'check in' or 'check out' software changes on an MVM client, these changes are sent to or received from an MVM daemon via a TCP/IP connection.

MVM Configuration Observer (MCO)

This process will run on the ACME company cloud. It will be responsible for reading global MVM settings and providing this information to interested MVM GUI and MVM Daemon applications.
When MVM Clients start up, they connect to the MCO, which provides it with a proxy object to the appropriate MD process.

MVM Daemon (MD)

MVM Daemons are server-side processes that perform version control operations on the MVM clients behalf. These daemons will run on ACME's file servers, and each MD is associated with a software account. Each software account stores software on its home directory, and individual files are version controlled using RCS.

Additional Information

Process Privileges

MCO processes run with Administrator privileges on the ACME clouds and MD processes run with the privileges of its associated software account.

It is assumed that if users can access a machine running an MVM Client then a user has permission to be interacting with MVM as that user. When the MVM Client runs, it obtains information about the running user from the operating system, and provides this information to MCO. MCO will determine if the user has associated with a software account and, if so, the right MD proxy object is returned to the MVM Client.

Connectivity between components

MVM was designed on the basis that components would be connected using DCOM.
In the past 12 months, this interface code has been migrated to use ZeroC's ICE framework, although the behaviour of the components and connectors remains the same.

MVM data formats

MVM clients rely on a number of different configuration files, but Midnight Engineering have stated that these are 'proprietary' and won't reveal anything about it.
MCO services read from an XML-based configuration file detailing software accounts and machine locations; this file is maintained by system administrators.
MD daemons work directly with RCS files. As an open source project, the structure of RCS files is well established. MD processes send file copies to, and receives modified files from MVM clients.

Questions

1. Evaluate the attack surface of MVM
2. Using appropriate security patterns, modify this software architecture to support authentication