This protocol is between metadata server (MDS), client (C), and key distribution server (KDS) for a file with root key K_R and user A. We assume that KDS is given private key $\boldsymbol{KR}_A$, which is paired with public key $\boldsymbol{KU}_A$.

The file in question is owned by the application A, and we want to allow the full access permission to the file only to the application A and the trusted KDS. In other words, both C and MDS are not allowed to access the entire file, since both C (i.e., a computation node in a supercomputer system) and MDS may be managed by a company that is different from the entity that is trying to run the application A on the supercomputer[1].

MDS provides the attributes of the file; i.e., the owner, permissions (modes), the key-hash parameter for the file (bs, d), permitted range (i, j) for a client C. The file attributes are generally open to public, meaning that MDS does not authenticate clients nor need to have clients' IDs. MDS does not need to distinguish KDS from C. MDS may provide storage service for the keys managed by KDS; MDS may hold in an extended attribute the master key K_R encrypted by KDS's public key KU_{KDS} or A's public key KU_A. In this way MDS cannot have K_R and hence cannot have the full access to the file. MDS can be implemented as the meta data server in Ceph or extended attributes (or "*forks*") in other file systems.

$$
\begin{aligned}
MDS &: \text{a meta data server.} \\
C &: \text{a client or a computation node.} \\
KDS &: \text{a key distribution server.} \\
A &: \text{a user or an application.} \\
\boldsymbol{KR}_A &: \text{the private key for } A. \\
\boldsymbol{KU}_A &: \text{the public key for } A. \\
(bs, d, i, j) &: \text{block size } bs, \text{ depth } d, \text{ range } i, j.
\end{aligned}
$$

Key request:

$$
\begin{aligned}
C \rightarrow MDS &: \text{open request} && (1) \\
MDS \rightarrow C &: \text{file attributes: } \{bs, d, E_{KU_A}(K_R)\} && (2) \\
C \rightarrow KDS &: \{E_{KU_A}(K_R), \text{range}(bs, d, i, j), \text{identification of } C\} && (3) \\
KDS \rightarrow C &: K_{i,j} && (4) \\
C \rightarrow OSD &: \text{file access (read/write)} && (5)
\end{aligned}
$$

- we trust that the system is not compromised. The owners and the process IDs, etc, are not faked in the system. So we don't have to worry about the authentication generally (particularly for MDS).
- However, in order to make KDS serve key calculation service for C, KDS needs to authenticate C. In this case the identification of C can be $E_{KR_C}(bs, d, i, j)$ (range request signed by C).
- For confidentiality, the entire message in $C \rightarrow KDS$ can be encrypted by KU_{KDS}.
- As a version of design model, private key for A is also managed by KDS, not by C. In this case, C does not have $\boldsymbol{KR}_A$, but KDS has.

[1] Hence, the MDS should not have the file's master key K_R.