DCMTK - Bug #1233

Access-control gap in dcmqrscp

2026-06-22 15:03 - Michael Onken

Status:	Closed	**Start date:**	2026-06-22
Priority:	Normal	**Due date:**	
Assignee:	Michael Onken	**% Done:**	0%
Category:		**Estimated time:**	0:00 hour
Target version:			
Module:	dcmqrdb	**Compiler:**	
Operating System:			

Description

As reported by Yuxiao Yan:

```
Tested version
---------------
- Upstream git, commit 5708ba6c (cloned 2026-06-11).
- Built normally; dcmqrscp run over loopback and driven with the DCMTK SCU
  tools (findscu/getscu/movescu) shipped in the same build.
- The relevant code is unchanged on the current main branch; line numbers
  below are as of 5708ba6c (the original storage-refusal logic has been in
  place for a long time).
```

Summary

The per-AE "Access" field in the configuration file is documented to accept "R", "RW", or "W" (dcmqrdb/docs/dcmqrcnf.txt:133). An operator can therefore configure a storage area as "W" (write-only) intending to permit ingestion (C-STORE) while blocking reads (query/retrieve).

The implementation only consults this field to decide whether to refuse **Storage** presentation contexts. It never uses it to refuse C-FIND, C-GET, or C-MOVE. As a result, a "write-only" (Access W) AE still negotiates and serves Query/Retrieve to any peer that passes the AETable peer check. The admin's intent ("W = no reads") is silently not honored for the read paths.

Note that the Access model is asymmetric: a read-only AE (Access R) correctly gets Storage contexts refused, but there is no corresponding gate that refuses the read paths for a write-only AE.

Impact and scope

- Attacker: a DICOM peer that is accepted by the called AE (i.e. it passes the
AETable peerInAETitle host/AE-title check; in many deployments the peer list
is "ANY"). This is a remote, **authorized** peer, not an unauthenticated one.
- Config: a common, documented configuration -- a storage area with Access "W".
No non-default command-line option is required.
- Effect: against the "write-only" AE the peer can * enumerate stored studies/series/instances and read their metadata via
C-FIND (patient name, IDs, study/series/SOP UIDs, etc.), and * retrieve the stored objects themselves via C-GET / C-MOVE.
So the practical impact is information disclosure (PHI) from an AE the
operator believed could only be written to -- an access-control gap, not
memory corruption.

This is config-policy enforcement, not a spoofing/auth bypass: the peer must still be authorized by the AETable. The defect is that "W" does not mean what the documentation and the operator expect it to mean for reads.

Steps to reproduce

Config (AETable excerpt) -- one write-only AE and a read-write control:

```
AETable BEGIN
  WRITEONLY  /path/db/WRITEONLY  W   (200, 1024mb) ANY
  RWSTORE    /path/db/RWSTORE    RW  (200, 1024mb) ANY
  AETable END
```

Start the server:

```
dcmqrscp -d --single-process -c qrscp.cfg 11112
```

Drive it (the WRITEONLY AE should, per its "W" access, refuse query/retrieve):

1. C-FIND against the write-only AE
   findscu -v -d -S -aec WRITEONLY -aet TESTSCU localhost 11112 \
   -k "0008,0052=STUDY" -k "0010,0010"

1. C-GET against the write-only AE
   getscu -v -d -S -aec WRITEONLY -aet TESTSCU localhost 11112 \
   -k "0008,0052=STUDY" -k "0020,000d=1.2.3"

1. C-MOVE against the write-only AE
   movescu -v -d -S -aec WRITEONLY -aet TESTSCU -aem DEST localhost 11112 \
   -k "0008,0052=STUDY" -k "0020,000d=1.2.3"

Observed (all three should have been refused, but were not):

```
FIND Study Root  -> ctx 1.2.840.10008.5.1.4.1.2.2.1  => ACCEPT
  GET  Study Root  -> ctx 1.2.840.10008.5.1.4.1.2.2.3  => ACCEPT
  MOVE Study Root  -> ctx 1.2.840.10008.5.1.4.1.2.2.2  => ACCEPT
```

and the C-FIND completed at the DIMSE level (not just context acceptance):

```
D: Message Type   : C-FIND RSP
  D: DIMSE Status   : 0x0000: Success: Matching is complete
```

For comparison the read-write AE (RWSTORE) accepts the same contexts, as expected -- so the write-only AE is behaving identically to the read-write one on the read paths.

Root cause (file:line, commit 5708ba6c)

In DcmQueryRetrieveSCP::negotiateAssociation() the only per-AE access check is for storage contexts:

```
dcmqrdb/libsrc/dcmqrsrv.cc:927-934
    /*
   * Refuse any "Storage" presentation contexts to non-writable
   * storage areas.
     */
    if (!config_->writableStorageArea(calledAETitle))
  {
      refuseAnyStorageContexts(assoc);
    }
```

The Query/Retrieve (non-storage) contexts are accepted just above/below this, gated only on global server options (supportStudyRoot_, disableGetSupport_, etc.), never on the called AE's access mode:

```
dcmqrdb/libsrc/dcmqrsrv.cc:908-912
    /*  accept any of the non-storage syntaxes */
    cond = ASC_acceptContextsWithPreferredTransferSyntaxes(
        assoc->params,
        (const char*)selectedNonStorageSyntaxes, numberOfSelectedNonStorageSyntaxes,
        (const char**)transferSyntaxes, numTransferSyntaxes);
```

And the config layer only implements a "writable" predicate -- there is no "readable" counterpart:

```
dcmqrdb/libsrc/dcmqrcnf.cc:1109-1116
    OFBool DcmQueryRetrieveConfig::writableStorageArea(const char *aeTitle) const
  {
        const char axs = getAccess((char)aeTitle);
        if (strcmp(axs, "RW")  0) return OFTrue;
        if (strcmp(axs, "WR")  0) return OFTrue;
        if (strcmp(axs, "W")  == 0) return OFTrue;
        return OFFalse;
    }
```

So "W" is treated as "writable" (correct for storage), but nothing maps "W" to "not readable" for the FIND/GET/MOVE paths.

Suggested fix

Mirror the existing storage gate for the read paths. Concretely:

1. Add a readable-access predicate to the config class, e.g.
OFBool DcmQueryRetrieveConfig::readableStorageArea(const char *aeTitle) const {
const char **axs = getAccess((char**)aeTitle);
return (strcmp(axs, "R") 0 || strcmp(axs, "RW") 0 || strcmp(axs, "WR") == 0);
}
(i.e. "R"/"RW"/"WR" are readable; "W" is not.)

2. In negotiateAssociation(), in the same place where storage contexts are
refused for non-writable AEs, refuse the Query/Retrieve abstract syntaxes
(FIND/GET/MOVE, patient-root / study-root / patient-study-only) when
!config_->readableStorageArea(calledAETitle) -- e.g. by excluding them from
selectedNonStorageSyntaxes for a non-readable AE, leaving Storage (and the
private shutdown class, if enabled) intact.

This makes "W" behave as the documentation implies (write-only: ingestion allowed, query/retrieve refused) and keeps "R" and "RW" unchanged.

If you would prefer not to change the enforcement semantics, an alternative would be to document explicitly that "Access W" does not restrict query/ retrieve and that read restriction must be enforced by other means -- but given that "R" already restricts storage, symmetric enforcement seems closer to the intent.

History

#1 - 2026-06-22 15:14 - Michael Onken

Closed by commit 696abc.

#2 - 2026-06-26 07:57 - Michael Onken

- Status changed from New to Closed