ARM Session

Dmitri Kondratiev

dkondr@bigfoot.com

Table of Contents

Abstract **3**
Overview of the Protocol **3**
New Session Announcement and Creation **3**
Joining Session **5**
Negotiating Sub-session **5**
Use Cases: DC Manager, ARM Session Manager and Retail Store Manager **5**

Abstract

This document describes ARM Session protocol, its implementation with ARM Session Manager, its role in CVM as well as inter operation with other components. ARM Session is a session management protocol that provides extension facilities for AA to fine-tune session parameters.

Overview of the Protocol

The role of ARM Session (ARMS) protocol is the complete control of ARM sessions described in a separate document. ARMS is message-oriented protocol used by ARM Session Manger (ARMSM).

ARMSM has a server component that is used by AA that wants to create ARM session. ARMSM has also a client component that is used by 'session clients' - AAs potentially interested in using newly announced sessions.

ARMSM Server sends session announcements to a well-known public group that session clients are listening. ARMSM Server also receives client join request from public group they all use for communicating messages to each other. To send 'join' request session client uses the client part of ARMSM.

ARMS Manager uses ARMS for the following session control functions:

- Announcement of a candidate session in the public group. Candidate sessions are those that may be created and activated providing there is at least one session client interested in the candidate session.
- Collection of join requests from session clients interested in the candidate session. Join requests are collected during announcement interval, after which this session announcement is closed.
- Activation of the session after session announcement is closed.
- Maintenance of client membership. Membership control here means monitoring of clients leaving the session while it is still active.
- Session termination announcements. When session terminates ARMS sends announcement to the public group.

New Session Announcement and Creation

This section describes announcement and creation of a new session by AA working together with ARMSM Server.

AA that wants to invite other AAs to participate in ARM session builds session announcement (SA) packet and requests ARMS manager to make an announcement. ARMSM multicast SA packet to well-known public announcement group that all interested clients has joined at boot-up.

SA packet includes:

- Unique Major ARM Session ID.
- Unique Minor ARM Session ID.
- Session Announcement Interval.
- VPN Parameters.
- Group address (class D) and port of the working group to be used for the session.
- Application Info. This is application-specific data which is defined by AA creating session. ARM sessions may be used by different AAs to participate in sessions

with different parameters. Application Info is used to store these parameters. For example, in TFTP case, this at least will include file name.

When session client decides to participate in new session it uses ARMSM Client to send 'join' request packet to ARMSM Server. Join request mirrors SA packet and includes all its fields as well as some new ones. Important new field is client IP that server uses for client accounting. Client may modify Application Info by changing and/or adding new fields to it. This mechanism allows communicate application-specific info between session creator AA and session client AA in order to fine-tune the session.

Join packet includes:

- Unique Major ARM Session ID.
- Unique Minor ARM Session ID.
- Session Announcement Interval.
- VPN Parameters.
- Client IP - IP of the client that joins the session.
- Group address (class D) and port of the working group to be used for the session.
- Application Info.

When ARMSM Server receives join request from some client it notifies session creator AA about this event sending it join packet as it is. Session creator now knows that at least one client is interested in his session and builds ARM session channel configured to use group address and port received in join packet.

When channel is built, session creator AA requests ARMSM Server to activate the session. AA also needs to know when session will be actually activated so it registers as a listener of 'session activation event' with ARMSM Server.

ARMSM Server waits for Join Request that may come to public announcement group during some 'announcement interval'. This interval is set by creator when it requests announcing the new session and may vary per session.

In reply to Join Request ARMSM Server send ACK Join packet to notify client that it has become a valid session member.

When 'announcement interval' expires ARMSM Server stops sending 'old' announcements to public group and sends activation event to session creator AA. This event includes session ID and working group multicast address and port that was originally set by ARMSM in session announcement packet.

Immediately after 'old' announcement is stopped ARMSM start sending new announcement to the public group. The new announcement has the same fields as the old announcement except for minor session ID and work group address/port. For these fields ARMSM generates new values. This new announcement is sent as long as announcement interval has not yet expired. Creator AA may request ARMSM Server to abort at any time all running sub-sessions with given Major Session ID.

Every time creator AA receives session 'activation event' it activates (and builds when needed) new ARM session channel and starts sending data to it. The mechanism of 'sub-session' announcements allows to solve the problem of 'late joiners' and gracefully scale transmission without overloading network with redundant data. Late joiners are clients that for some reason join session late. Joining late may happen because of missing session announcement (SA) packet by client, missing join packet by ARMSM Server or because of the other constraints that client may have locally.

Every 'sub-session' has the same Major Session ID and different Minor Session ID. AA uses unique channel instances for every 'sub-session' / Minor Session ID, sending data to different work groups. In other words, in every work group data is sent to unique multicast address/ port combination. As a result data already sent to early joiners and that late joiners is still missing is only sent to those who need it. The

state of the sent and unsent data is maintained automatically in every instance of the channel.

The cycle of generating new 'sub-session' announcement is fully controlled by session creator AA. In addition to aborting session completely (see above) it may request ARMSM to stop generating announcements. This may be needed if there is a need to restrict resources used by AA for session creation.

Joining Session

When the client part of ARMSM receives Session Announcement (SA) packet it sends it to a listener AA interested in session announcements.

Listening AA may decide to join the new session. In this case AA requests ARMSM Client to start sending Join Request to ARMSM Server. ARMSM Client sends join request repeatedly until it gets ACK Join answer from ARMSM Server. ACK Join answer confirms that server has accepted this client as a valid member of the working group with given Major Session ID and Minor Session ID.

ACK Join request includes:

- Unique Major ARM Session ID.
- Unique Minor ARM Session ID.
- Session Announcement Interval.
- VPN Parameters.
- Client IP.
- Group address (class D) and port of the working group to be used for the session.
- Application Info.

When ARMSM Client receives ACK Join packet it passes it to AA listener that creates ARM channel with given group address and port as specified in ACK Join packet. When channel is built session client AA starts receiving the data.

Negotiating Sub-session

There may be situations when Join and ACK Join packets are getting lost or come out of order. For session client this leads to confusion: what sub-session / work group to join ?

To avoid sub-session confusion the following strategy is used:

- When ARMSM Server starts announcing new sub-session it immediately stops sending ACK Join-s for the previous sub-session.
- If ARMSM Server gets Join Request from a client that already has joined this session it removes client from older sub-session and sends ACK Join to the client with parameters of the current sub-session announcement.
- When ARMSM Client waiting for ACK Join gets new sub-session announcements it stops sending Join Requests for an older sub-session and starts sending Join Requests for newer sub-session.
- When ARMSM Client gets ACK Join for an older sub-session - it ignores it and continues joining newer sub-session.

Use Cases: DC Manager, ARM Session Manager and Retail Store Manager

The following diagram illustrates main uses cases of managing different sessions in CVM with specialised managers. Managers are AA that provide protocol and task specific manager channels. This document describes three managers: ARM Session Manager working together with Data Centre and Retail Store Managers.

BSG1

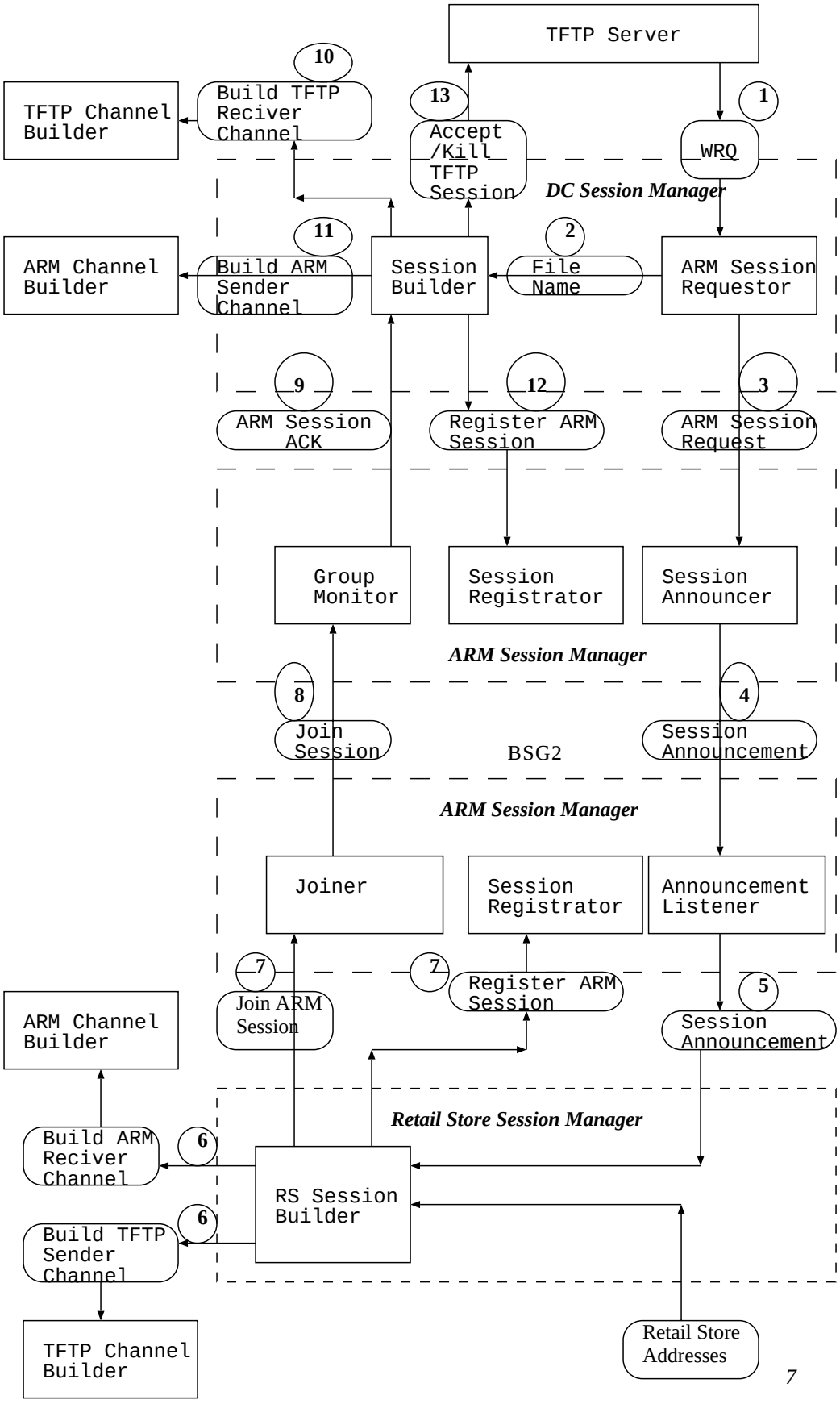

Figure 1. Main Interactions

Operation

Abbreviations

- DC - data centre, DCSM - Data Centre Session Manager
- SR - Session Requestor
- SReg - Session Registrartor
- SB - Session Builder
- SM - Session Manager
- SA - Session Announcement
- RS - Retail Store
- RSSM - Retail Store Session Manager
- AL - Announcement Listener
-
-

Main Algorithm

- BSG1: DC Session Manager (DCSM) registers ARM Session Requestor as a listener for TFTP server WRQ events.
- BSG1: TFTP server receives WRQ request (1) from Data Centre. DCSM ARM Session Requestor (SR) extracts file name from WRQ request and checks if there is an open session associated with this file. If so SR ignores WRQ and nothing else is done in this step. Otherwise SR sends file name to DCSM Session Builder (SB). SB uses file names to maintain a hash of pending sessions. Then SR requests new ARM session (3) passing TFTP file name as application info parameter associated with this ARM session request.
- BSG1: ARM Session Announcer generates unique ARM session number and builds Session Announcement (SA) packet. This packet includes ARM session number and application info. Application info is passed as a parameter to ARM session request. In this case info includes file name sent by DCSM ARM Session Requestor.
- BSG1: ARM SM sends SA packet to ARM SM on the second level BSG2 (4).
- BSG2: ARM Announcement Listener (AL) receives SA packet from BSG1. AL sends SA packet to registered listeners - in this case to RS Session Builder (5).
- BSG2: RS Session Builder builds ARM Receiver Channel and TFTP Sender Channels (6).
- BSG2: RS Session Builder sends Join ARM Session Request to ARM SM Joiner (7).
- BSG2: RS Session Builder sends Register ARM Session (RAS) packet to ARM Session Registrator (SReg).
- BSG2: ARM SM Joiner builds "Join ARM Session" packet and sends it to ARM SM on first-level BSG (8). "Join ARM Session" packet includes ARM session number and application info sent in SA packet from BSG1. Thus both of these parameters make a round trip from / to BSG1.
- BSG1: ARM SM Group Monitor receives join packet and notifies all listeners interested in group membership events. In our case this listener is DCSM Session

Builder. Listener receives ARM Session ACK packet which includes session number and application info from SA packet.

-
-