Errata to SyncML Synchronization Protocol

Specification version: 1.0

Specification date: 2000-12-07

SyncML Initiative

The following companies are Sponsors of the SyncML initiative:

Ericsson
IBM
Lotus
Matsushita Communications Industrial Co., Ltd.
Motorola
Nokia
Palm, Inc.
Psion
Starfish Software

Copyright Notice

1 Formatting Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interoperated as described in [RFC 2119].

1.1 Errata Type Classifications

The errata types are classified according to the following scheme:

CLARIFICATION: Textual enhancement that provides a clearer explanation of a specification item without changing any behavior.

CORRECTION: A modification that obsoletes some items in the current published specification.

PROBLEM: A known problem for which an erratum has yet to be proposed.

2 Requirement for CmdID element within Status

2.1 Problem

The SyncML DTD defines that the CmdID element is mandatory within the Status element. However, the Synchronization protocol specification defines that the CmdID element must not be used within the Status element.

2.2 Solution

Chapter 4.2: Remove the last bullet saying "The CmdID element inside the Status MUST NOT be used. This rule applies for all status elements used by this protocol" of the requirement regarding the Status element (requirement #2).

Add the CmdID element into the Status element in all the examples in the specification.

2.2.1 Other specifications/erratas affected

None.

3 Content of Item in Result Alert

3.1 Problem

In the example in Chapter 2.11.2 the Target and Source elements within the Item element incorrectly specify the target and source databases.

3.2 Solution

Chapter 2.11.2: Change the Target and Source elements inside the Item element within the Alert command in the example to point to the client and server devices, as defined below.

```
<SyncML>
  <SyncHdr>
    <VerDTD>1.0</VerDTD>
    <VerProto>SyncML/1.0</VerProto>
    <SessionID>1</SessionID>
    <MsgID>3</MsgID>
    <Target><LocURI>http://www.syncml.org/sync-server</LocURI></Target>
    <Source><LocURI>IMEI:493005100592800</LocURI></Source>
  </SyncHdr>
  <SyncBody>
    <Alert>
      <CmdID>1</CmdID>
      <Data>221</Data>
      <Item>
        <Target><LocURI>http://www.syncml.org/sync-server</LocURI></Target>
        <Source><LocURI>IMEI:493005100592800</LocURI></Source>
      </Item>
    </Alert>
  </SyncBody>
</SyncML>
```

3.2.1 Other specifications/erratas affected

None.

4 Changing authentication type during a session

4.1 Problem

The specification is unclear whether the authentication type can be changed (swapped) during the session on a specific security layer.

4.2 Solution

Chapter 3.1: Add the following paragraph to the end of the section: The authentication type for a security layer MUST be kept same for the whole session."

4.2.1 Other specifications/erratas affected

None.

5 Slow sync when no separate initialization

5.1 Problem

The specification is unclear what happens when the server requests the slow sync when doing sync without a separate initialization phase.

5.2 Solution

Need to clarify that the client needs to send the all updates in the next message because the slow sync was requested.

Chapter 5.5: Modify the fourth paragraph of the section to "If the client or the server needs to initiate the slow sync after receiving the alert for the normal synchronization, they need to send back an error status for that Alert in addition the slow sync alert. The error code, which is used in this case, MUST be 508 (Refresh required). If the client has not used a separate synchronization initialization, as specified in Chapter 2.10, it MUST send all updates in the next message to the server after receiving the error status and the Alert for a slow sync. "

5.2.1 Other specifications/erratas affected

None.

6 Support of RespURI element

6.1 Problem

The Representation protocol specification defines that the client and server must be able to receive and send the RespURI element, respectively. The Sync Protocol specification says, the support is completely optional.

6.2 Solution

Chapter 2.6.1.1: Change the text "This protocol does not require the support of the RespURI element. Either the support of the re-direction status codes (3XX) is not required." to "This protocol requires that the devices support receiving the RespURI element as specified in the SyncML Representation Protocol specification, but the support of the re-direction status codes (3XX) is not required."

6.2.1 Other specifications/erratas affected

SyncML Representation protocol, see also the Errata for SyncML Representation Protocol.

7 Multiple messages per package functionality

7.1 Problem

Clarification needed on Multiple messages per package functionality.

7.2 Solution

Change the text in Chapter 2.9 to look like this:

This protocol provides the functionality to transfer one SyncML package in multiple SyncML messages. This may be necessary if one SyncML package is too large to be transferred in one SyncML message. This limitation may be caused e.g., by the transport protocol or by the limitations of a small footprint device.

If a SyncML package is transferred in multiple SyncML messages, the last message in the package MUST include the Final element (See SyncML Representation protocol.). Other messages belonging to the package MUST NOT include the Final element. The Final element must only be included when all necessary commands belonging to a specific package have been sent. The final element must not be included if the other end has not closed the preceding package. E.g., if the server is still sending the package #4 to the client, the client must not close the package #5 prior to receiving the last message belonging to the package #4. The exclusion of the Final element must not be used to indicate that a logical phase is not completed if an error occurs.

If a device receives a message in which the Final flag is missing and a Sync element for a database is included, the device MUST be able to handle the case that in the next message, there is another Sync element for the same database.

The device, which receives the SyncML package containing multiple messages, MUST be able to ask more messages. This happens by sending an Alert command with a specific alert code, 222 back to the originator of the package, or if there are other SyncML commands to be sent as a response, the Alert command with the 222 alert code can be omitted. After receiving the message containing the Final element, the Alert command MUST NOT be used anymore.

More messages may not be desired if errors, which prevent the continuation of synchronization, have occurred.

The receiver of a package may start to send its next package at the same time when asking more messages from the originator if this makes sense. Thus, in Chapters 3-7, it is specified which commands or elements are allowed to be sent before receiving the final message belonging to a package.

Below, there is depicted an example that the sync client is sending Package #3 in multiple messages (2 messages) and the server also sends Package #4 in multiple messages (2 messages).

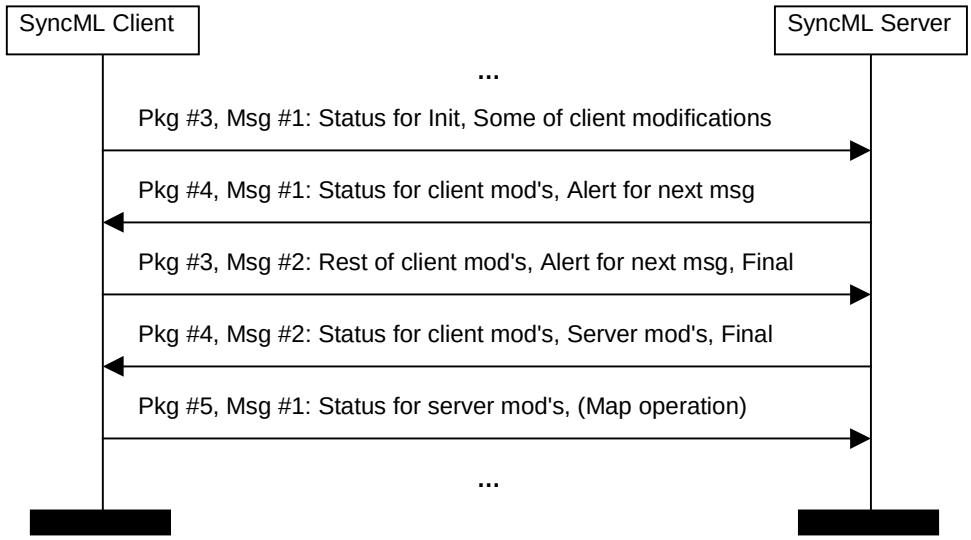

Figure 1 Example of Sending Multiple Messages in a Package

7.2.1 Other specifications/erratas affected

None.

8 MD5 digest access authentication example

8.1 Problem

The example in Chapter 3.5.2 needs to updated to include the UserID in the LocName element (in the Source element) of SyncHdr.

8.2 Solution

Chapter 3.5.2: Add the LocName element into the Pkg#1 (with credentials) in the example:

```
<SyncML>
  <SyncHdr>
    <VerDTD>1.0</VerDTD>
    <VerProto>SyncML/1.0</VerProto>
    <SessionID>1</SessionID>
    <MsgID>2</MsgID>
    <Target><LocURI>http://www.syncml.org/sync-server</LocURI></Target>
    <Source>
```

```
      <LocURI>IMEI:493005100592800</LocURI>
      <LocName>Bruce2</LocName> <!-- userId -->
    </Source>
    <Cred>
      <Meta><Type xmlns='syncml:metinf'>syncml:auth-md5</Type></Meta>
      <Data>NTI2OTJhMDAwNjYxODkwYmQ3NWUxN2RhN2ZmYmJlMzk=</Data>
      <!-- Base64 coded MD5 digest of "Bruce2:OhBehave:Nonce" -->
    </Cred>
  </SyncHdr>
  <SyncBody>
  ...
  </SyncBody>
</SyncML>
```

8.2.1 Other specifications/erratas affected

None.

9 Authentication failure

9.1 Problem

Current Sync protocol specification doesn't specify any requirements for authentication failure. This may lead into a situation where the response contains not only a status code indicating that the authentication failed, but also commands like Put and Get.

9.2 Solution

It needs to be clarified in chapter 3.1 that in case of authentication failure (either the userid and/or password was wrong or authentication was required) requirements are:

- The response message indicating the authentication failure on server layer (see chapter 3.3) must contain only Status commands (i.e. Put, Get etc. commands MUST NOT be specified in the response)
- In case the session is continued, the next message containing the proper credentials MUST contain a Status for the SyncHdr, MUST have the same SessionID than the previous messages and the message MUST be sent to the RespURI, if it was specified in the response indicating the authentication failure.

9.2.1 Other specifications/erratas affected

None.

10 Successful Sync

10.1 Problem

Currently the SyncML synchronization protocol specification only defines, that sync anchors need to be updated after the sync has been finished. However this definition is quite vague, and it needs to be clarified when the synchronization has been ended, so that the anchors must be updated.

We need to define that a sync has been finished after the transport level directly below SyncML level (e.g. HTTP, WSP, OBEX) has been closed properly. In HTTP case this would mean that all of the HTTP-requests have gotten a response back from the server. It is necessary to define that the transport need to be closed before the sync is considered to be finished, otherwise we might end up in a situation in which the transport was "broken" during the session, but still the other end (e.g. a server) would determine that the session was successful.

An example: the client has sent its modifications to the server -> the server sends its modifications to the client -> the client needs to send a response to the server's sync and replace commands, but no other commands in this package (like maps) -> server has no SyncML data to be sent to the client and it closes the connection (i.e. doesn't send the HTTP 200 response). Now, the client can't tell whether the server was able to receive the status commands and it doesn't update its anchors, but the server believes that the session ended ok, and the server updates its anchors. This will lead into an unnecessary slow sync next time between the client and the server.

10.2 Solution

Chapter 2.2.1, add the paragraph below to the end of chapter:

The synchronization session is finished after a device is not going to send and is not expecting to receive any SyncML messages from other device, and the synchronization was successfull on the Sync command level (i.e. no other than 200-class statuses has been returned for Sync commands). Also the transport level (directly under SyncML level) communication has to be properly ended before synchronization can be seen as finished. If the communication between synchronizing devices is not ended properly according to transport level specification, devices MUST NOT update their sync anchors.

10.2.1 Other specifications/erratas affected

None.

11 Clarification on Slow Sync

11.1 Problem

The Slow Sync description does not include text describing what to do if the server sends a Sync Alert containing a sync anchor that disagrees with what the client remembers.

11.2 Solution

Insert a new paragraph #4 to the protocol document in section 5.5 with the following text:

"After the server has sent the Sync Alert, and if the client does not agree with the sync anchor in that Alert, then the Client MUST start a slow sync. This is done by sending back a Status on that Alert with Refresh Required. In this same message, the client should start the slow sync. In this case, the client MUST NOT send another Alert to start the slow sync. Note that it is not necessary for the client to compare the sync anchor from the server."

11.2.1 Other specifications/erratas affected

None.

12 Server required to send Map Acknowledgment

12.1 Problem

The Server is required to send a message back to the client regardless of whether or not there is any data to send. Section 5.4 has the following sentence: "This acknowledgement is not sent back to the client if there were no Map operations in last package from the client to the server." That sentence contradicts the requirement.

12.2 Solution

Change the sentence to read:

"This acknowledgement is sent back to the client even if there were no Map operations in last package from the client to the server."

12.2.1 Other specifications/erratas affected

None.

13 References

[RFC 2119] Key words for use in RFCs to Indicate Requirement Levels, IETF.