Duplicate Remittances

Ref: c:\users\gareth\documents\gij\work - offline\rem issue\duplicate rems.docx
Author: Gareth I Jenkins
Date: 19/11/2015 14:34:00

1. Introduction

The purpose of this note is to describe the problem identified at the Dalmellington Outreach Service on 8th October 2015. It also looks at other similar occurrences and what could be done to identify any other similar problems and their occurrences.

2. The Original Problem

A Postmistress who runs a Core and Outreach branch remitted £8,000 out of the Core branch and then attempted to remit the cash into the Outreach Branch. However when the cash was remitted into the Outreach Branch the system repeated the inward remittance transaction 3 times, thus remitting in a total value of £32,000. This was £24,000 more than the actual cash being remitted and so resulted in the system at the Outreach Branch showing a deficit of £24,000.

The Postmistress raised a call and subsequently reported the incident to CWU. As a result the whole issue has become very political and is bringing the integrity of Horizon into question.

It should be noted that receipts were generated for all the remittance transactions and that they are all clearly visible in the Transaction Log (which is viewable locally) and also in the Audit Trail (which is used as the basis for any evidence of the system's behaviour).

Fujitsu have investigated the problem and can reproduce the problem. A fix for it has also been identified. These are described below.

2.1 <u>How the problem occurred</u>

The following describes the sequence of events that occurred for this problem to occur. All these activities need to be carried out on the same Horizon terminal in the order described:

1. A User Logs On to a Horizon terminal
2. As part of the Log On process the Log In checks identify an activity that needs to occur as part of the Log On process (this is done by what is called a Post Log On script in Horizon). The activity is likely to be either a missing Cash

[1] Each Stock Unit is required to make a Cash Declaration at the end of each day that it is used. Should no such Cash Declaration have been made, then the first User attached to that Stock Unit is prompted to make a Cash Declaration as part of the Log On process the following day

[2] If a Branch has an external device such as a PayStation or a Camelot terminal, then the first user to Log On in the morning is required to acknowledge the acceptance of a transaction representing the amount of cash taken on that external device as part of the Log On process.

Declaration[1] or processing a Transaction Acceptance[2]. For the purpose of this description we will take the case of a missing Cash Declaration.

3. The User goes part way through the process, but does not complete it. For example a new Cash Declaration is made, but the User pauses at the screen where they are asked if they wish to Print or Preview the Declaration
4. The User then leaves the terminal in this state for 15 minutes.
5. Horizon then locks the terminal as being inactive.
6. Either another User goes to the Locked Terminal and forces a Log Out, or the terminal remains idle for a further 59 minutes. In either case there is a forced Log Out of the session at the terminal.
7. Any User (either the original User or some other User) Logs On at the terminal. Note that this could be some time later, but must be on the same day as the system is closed down and restarted overnight.
8. Immediately after this Log On the first transaction carried out by the User is a Manual[3] Cash Remittance (this is done by the Pouch Delivery script).
9. When the Remittance is complete and all the receipts have been printed, the last screen asks the user to press Enter to complete the Remittance process.
10. However, rather than completing the process, the effect of the Enter button is to repeat the recording of the remittance transaction and the printing of the Remittance receipt then redisplaying the screen asking the User to press Enter.
11. Each time the User presses Enter the remittance is repeated. However if the User presses Cancel (instead of Enter) then the system comes out of the Remittance process and all is well again.

Please note that this description is significantly different to that carried in Computer Weekly. This scenario is easily repeatable and it is not an intermittent issue.

2.2 The Actual Problem

There are actually 2 separate issues here, and it is the combination of the two issues that leads to the scenario described above:

1. The Forced Log Out described at step 6 above doesn't correctly close down the Post Log On script. This leaves the script on the "stack" of incomplete processes

[1] Each Stock Unit is required to make a Cash Declaration at the end of each day that it is used. Should no such Cash Declaration have been made, then the first User attached to that Stock Unit is prompted to make a Cash Declaration as part of the Log On process the following day

[2] If a Branch has an external device such as a PayStation or a Camelot terminal, then the first user to Log On in the morning is required to acknowledge the acceptance of a transaction representing the amount of cash taken on that external device as part of the Log On process.

[3] Note that normally a Cash Remittance is an Automatic Remittance taking the value of the cash in the pouch from records generated at the Cash Centre when the pouch is packed. However Branch to Branch Remittances (which are used when transferring money between Core and Outreach Branches) don't go via the Cash Centres and so there is no record of the pouch content in Horizon which means that the value of the pouch content has to be keyed in manually in the receiving branch.

2. The Pouch Delivery script when it thinks it has completed at step 9 above doesn't explicitly finish. It relies on a mechanism which checks the stack of incomplete processes to see if it is complete. Due to the fact that the stack is not empty (following the first problem) it thinks it has not finished and as a result attempts to repeat the last part of the script, which in this case is to record the remittance transactions and print the receipts.

It is likely that this problem has been present in Horizon since Horizon Online went Live during 2010. However this particular problem would not have been present on the old Riposte-based Horizon system that was running until 2010.

2.3 Proposed Fix

The proposal is to fix the first part of the problem described above, ie to ensure that a Forced Log Out correctly clears down the stack of incomplete processes.

3. Related Problems

It appears that there have been similar issues in the past where the mechanism associated with checking the incomplete stack has resulted in duplicate transactions. Specifically there was a case of a Session being Resumed after being Suspended, and the value of the suspended session was duplicated in the Basket of the resumed session.[4]

It is also worth noting that both of the problems described in section 2.2 are specific instances of what could be a class of problems. There may be other scenarios that result in the stack being left in a corrupt state and there may be other transactions which repeat themselves when run against a corrupted stack.[5]

However in all cases the Transaction Log and Audit will show exactly what has happened.

4. Other Examples of this Scenario

Once this problem was diagnosed, SSC looked to see if there were other examples of this scenario. The simplest way to look for them is to search for multiple Remittance transitions for the same Pouch ID. This can be done by querying the BRDB_RX_EPOSS_TRANSACTIONS table in BRSS. However data is only retained in this table for 60 days so the search is restricted to the last 2 months. This check was carried out in October after the original problem was diagnosed and 4 other occurrences were identified. However in all 4 cases actions had been taken to resolve the issue using business processes.[6]

As a result of further discussions with SSC, it was realised that analysis of the BLE files (sent from Horizon to POL SAP each night) would also identify such problems. BLE files are stored for 6 months and so a check has been carried out back to mid-

[4] I don't have the full details such as Peak references or timescales for this issue. However I do recall it happening.

[5] On the other hand these may well be the only examples of such cases!

[6] There are 2 ways to do this: Either Remit Out the duplicate value or request a Transaction Correction (the branches appear to have thought that the amount had been miss-keyed and hence requested a TC to correct the miss-keying error)

May and 4 further occurrences of the issue have been identified.[7] Note that in one of these cases Currency was also being remitted and duplicate Currency values were also recorded.

5. Further Analysis

There are two areas where further analysis should be considered:

1. Checking old data for the precise scope of the issue
2. Checking the code for any other transactions that may fall into one of the classes of failure described in section 2.2.
3. Other possible fixes

These are discussed further below.

5.1 Checking Old Data

There are a number of approaches to this depending on exactly what level of checking is required. The two basic approaches are as follows:

1. As the monitoring of Cash Movements is fairly key to Post Office Ltd's business, then presumably they have some mechanism for correlating manual remittances between Core and Outreach branches.[8] This problem should easily be visible to such a correlation process and so if it is in place this should identify all occurrences of such an issue.
2. Restrict the checks to those Branches that have gone to mediation. As part of the mediation process, data has already been retrieved relating to the periods is question[9] and so it should be fairly simple to check that data for examples of this scenario.
3. Use the BLE files as a basis for the analysis. The main issue here is just the volume of data. It is understood that BLE files have been archived on the Audit Storage. This could be for 18 months or 7 years.[10] Such data could then be retrieved and then analysed in the same way that SSC analysed the data from the last 6 months. Note that the scripts used do not handle cases where only currency was remitted between the Core and Outreach Branches.[11] It is estimated that the analysis of 6 years' worth of data could be carried out in less

[7] Note that as part of this exercise it was found that duplicate Pouch Ids were identified in BLE files which do not relate to this issue. They were clearly distinguishable as having taken place in different Branches and also being for different amounts. However my understanding was that Pouch IDs were supposed to be unique and so this may be worth pointing out to Post Office Ltd as it may indicate some other issues.

[8] This needs to be confirmed with FSC. Note that such transactions do NOT get posted into the same GL account by the Horizon interface to POL SAP, and so this will not be handled by the Auto-matching mechanism used to check Cash between Branches and the Cash Centres. However there may be a manual correlation of such transactions between the two relevant accounts.

[9] However it is not clear if Fujitsu still has copies of this data or whether it was just passed over to Post Office Ltd.

[10] If it is only 6 months, then perhaps this isn't as useful as originally thought. 7 years certainly takes us back to the start of Horizon Online in 2010.

[11] It is not easy to extend them to do so.

than a man-week. However this does not include the time taken to retrieve the data.[12]

Note also that as manual Remittances are primarily (though not exclusively) used for Branch to Branch remittances which can only take place in Core and Outreach Branches[13], any search could be restricted to such Branches.

5.2 Checking for Other Failing Transactions

This is probably the key question, namely “What other transactions could be duplicated?”. Identifying all such transactions is likely to be very difficult and require a lot of human resources to carry out any such investigation. One approach that has been suggested is as follows:

1. Create an environment in which a corrupt stack exists.
2. Run each of the automated Counter Regression tests in this environment
3. Check the results and see if any of them behave differently to the way in which they are expected to behave.

However, this may not be as straightforward as it initially sounds. This is because:

1. Creating a corrupt environment may not be easy
2. Currently the automated regression tests are designed to be run as a single sequence of tests where many transactions are run one after another.
3. The majority of transaction force the stack to be cleared when they complete, so this means that it is only the first transaction in each regression test script that will actually execute on the corrupt environment.
4. Splitting the regression tests into individual transactions would require a lot of effort.

This could be considered further.

5.3 Other Fixes

The key to this problem is the use of a stack to control the outstanding work within a script and the problems encountered when using this stack to determine if a script has completed or not. It would appear that this area of the counter design has caused issues in the past and so it may be worth considering some radical changes in this area. Attempting to re-implement this at this stage in the Horizon lifecycle is not realistic. However it may be worth considering if there are points in the lifecycle of the CBA where the stack can be safely assumed to be empty and at such points to ensure that it is.[14] For example in the scenario described here, it should be safe at Log On to assume that there are no outstanding scripts, so a check at this point that the stack is empty

[12] There are about 64 BLE every day so for 6 years this is of the order of 150,000 files. Normally files are retrieved individually, but it may be possible to script a bulk extract. This would need to be investigated by the Audit team.

[13] This is probably policed by Reference Data. That needs to be confirmed.

[14] This is something that occurred to me as I was writing this note, and so I have not discussed it with the CBA designer / implementers and so it may not be appropriate.

(and clearing it and raising an alert if it isn't) would have avoided this issue. Returning to the menu system may well be another such point.

This needs further consideration.

6. Effect of a Forced Log Out on the Stack

Although this is nothing to do with this specific issue, Post Office Ltd have asked "what happens to an item in the stack when the user is timed out/force log off". The situation is as follows:

1. If the terminal times out due to inactivity[15] it becomes Locked[16]. This means that either the **same** User must supply their password again to unlock the terminal or **another** User can enter their own Username and Password and force a Log Off of the original User.
2. If a terminal remains locked for 59 minutes, then the system Forces a Log Out of the original User.

When a Forced Log Out occurs (either due to 74 minutes of inactivity or being forced by another User on a Locked Terminal), then the current Basket is checked. There are a number of Business Rules relating to the type of Basket. In simplistic terms they boil down to:

- If the Basket is a Customer Basket, then it should be settled to cash (as if Fast Cash had been selected)
- If the Basket relates to any Back Office function, then the Basket is discarded.

This is a slight simplification and there are a few added complexities relating to Reversals and Sessions that have been Resumed after Suspension, but the above covers most cases.

[15] Configured to happen after 15 minutes of inactivity

[16] It is also possible for a User to explicitly Lock a Terminal. However I'm not sure if this is allowed when there is something in the Basket. The effect of explicitly Locking the terminal is the same as that of a terminal being locked due to inactivity.