FUJ00087231

Database Security in Horizon Online

Ref: Database Securityv0.2
Author: Gareth I Jenkins
Date: 14/03/2017 15:58:00
Version: 0.2

1. Introduction

The purpose of this note is to explore how, in theory, a "super user" could modify the Horizon Online Audit Trails. It also goes on to discuss how this should be detected. This is covered in Section 2. Section 3 then considers the equivalent mechanisms in the old, Riposte based, Horizon system. Finally, section 4 considers things from the postmaster's viewpoint.

In conclusion, it should be evident, that although such attacks are theoretically possible, they would be very difficult and would be detected. Also there is no benefit to any person attempting to attack the system in this way.

2. Horizon Online

Section 2.1 describes how the Horizon Audit Trail is generated and secured. Section 2.2 then explores how, in theory, a portion of the Audit Trail could be deliberately replaced by a Super User. Finally, section 2.3 discusses how such changes could be detected.

2.1 How the Horizon Online Message Log is Generated and Secured

The Horizon Online Message Log is primarily a log of all auditable messages sent from the Horizon Counter to the Horizon Data Centre where they are processed by the Branch Access Layer (BAL) which in turn updates the Branch Database (BRDB).

> *Note, that not all messages sent from the Counter are audited. However any that could impact on the Branch accounts should be audited. Each message sent from the Counter to the Data Centre indicates whether or not it is to be audited (ie the decision is part of the counter application and not the BAL).*
>
> *The BAL configuration also decides which messages pass through the Audit Filter (which does the auditing).*
>
> *It is part of the system's design to ensure that the counter and BAL configurations are consistent in this respect.*
>
> *Specifically, messages have jsns if and only if they are auditable and so the checks on jsn sequences described below ensure the completeness of the audit trail.*

Each Auditable message sent from the counter includes a "digital signature" generated using the counter's Private Key. This key is generated by the counter as part of the Log On process. The corresponding Public Key is included in the Log On message sent from the counter to the BAL. Unlike other messages in the Message Log, the BAL adds a wrapper to this Log On message which includes a further digital signature (of the entire message including the counter's digital signature and the counter's Public

FUJ00087231

Key) generated by the BAL, using the BAL Private Key which is obtained from the NPS Key Store by the BAL at start-up.

All auditable messages are written to a single table within the BRDB know as the "Message Log". Each day (at some point after 1am) the previous day's Message Log is written to a number of files which are then passed to the Audit system which then "seals" each file and stores them until they are retrieved (if they ever are) or deleted. Note that each file will include records from a number of different Branches and there may be multiple files for a single day containing the records for a specific Branch.

Deletion of Audit records is currently suspended. They should be deleted after 7 years, but deletion was switched off sometime in 2014 (I think). Therefore all audit records since Horizon Online went Live in 2010 should be available.

This seal is cryptographically generated and is based on the entire contents of the Audit File. Any subsequent change to the contents would then invalidate the seal. The seal is held in a seals database separate from the Audit Data. A feature of the Audit System is that data cannot be amended or deleted until the pre-defined "Purge Date". Also Super Users do **not** have access to the Audit data.

All updates to the BRDB will be based on the information held in the auditable message and the accounts (both as seen in the Branch and also as passed to Post Office Ltd's back end accounting systems) are based on this information (and not on the actual auditable message). This means that in order to corrupt the Branch accounts, it is necessary to corrupt a number of different records within the BRDB and not necessarily the Message Log. However any evidence provided by Post Office Ltd is based on the Message Log – hence the need to corrupt the Message Log as well.

It is asserted that by going back to the audited data (ie the Message Log) sent from the Counter to the BRDB, then all Transactions that that counter carried out and their implications on the Branch Accounts can be re-calculated and compared with the reports produced by Horizon based on the other data held in the BRDB and Post Office Ltd's back end systems. This would enable any corruption of the data used to create the Branch reports to be detected from examination of the Message Log.

When audit data from the Message Log is retrieved for whatever reason, a number of checks are carried out to ensure the completeness and integrity of that data. These checks are:

- Each entire Audit File is checked to ensure that the digital seal stored at the time the Audit was produced (ie the day after the transactions took place) is valid.

Normally a Data retrieval will be for a number of days and so a number of Audit files will need to be retrieved.

- The data for the Branch in question is then filtered out from these audit files and checks are then carried out on a counter by counter basis as described below for the period of the extract:
    - No part of the Message Log is missing or duplicated. Each auditable message sent from the counter to the BAL incudes a unique sequence number (the Journal Sequence Number or jsn). The audit records for

FUJ00087231

any counter over a period of time should have no missing or duplicate jsns. The standard Audit Extracts into Excel include a report indicating that this check has been successfully carried out.

- The message audited as part of the Log On process, is checked and the Digital Signature generated by the BAL is checked by using the BAL's Public Key (which is known to the Audit System). This shows that this message was signed by an application which had access to the BAL's Private Key. This then provides access to the Counter's Public Key for that Log On Session (as this is included by the message audited by the BAL and was signed by the BAL's private key).
- All subsequent messages sent from the counter to the BAL during that Log On Session are then checked to ensure that their Digital Signatures are correct (using the Counter's Public Key obtained from the Log On message)

2.2 Replacing the Message Log

In theory, a Super User, could amend the Message Log for one or more Counters in one or more Branches. The following describes what would be required to replace the Message Log for a single counter in a single branch. This process could be repeated for multiple counters / branches if required.

1. The work would need to be completed before 1am the following day (since the Message Log is extracted from BRDB at some point after 1am each night and the data is then sealed and held in the Audit Server
2. The entire Message Log associated with a Log On Session that is to be corrupted would need to be replaced

This is because it is not possible to obtain the counter's Private Key and so a new one would need to be generated as described below.

3. The records being replaced would have to be in one-to-one correspondence to the original records otherwise there would be gaps or duplicates in the sequence of jsns which would then be detected as part of the Audit Retrieval process.
4. An application would need to be run by the Super Users in order to correctly construct the revised Audit Records
5. This application would need to generate a Private / Public key pair similar to the one originally generated by the counter. Called an "Attack Counter key" in the rest of the document
6. The application would need to generate a Public Key certificate for the Public Key

Andy T suggests that this doesn't happen so perhaps we remove this step.

7. The application would need to have access to the BAL's Private Key. Since this is stored in the Key Store which is an Oracle Database running on the NPS,

FUJ00087231

then it is assumed that a Super User would be able to read this value and make it available to the application. This would then enable the application to generate a Log On Message Log message containing the fake Counter Public Key and to sign it using the genuine BAL Private Key.

8. All subsequent messages for the session would then need to be amended as required and then re-signed using the attack Counter Private Key generated at step 5.

9. Having constructed all these false Message Log messages, then the Super User would need to delete all the genuine messages from the Message Log in BRDB and replace them with the false messages.

Or this could be done just by updating the rows with the new data. *Note that the table is designed to be appended to at all times. The impact of deleting or updating rows would have a significant impact on the performance of Oracle and this alone may be sufficient to prevent this form of attack.*

10. Note that as stated earlier, corrupting the Message Log in this way has no impact whatsoever on the Branch Accounts, since these never refer to the Message Log. The Branch Accounts are based on copies of some of the data held in the Message Log being stored in "working tables" within the BRDB. Clearly any application that is capable of corrupting the Message Log in BRDB would also be capable of updating (ie corrupting) the data used to calculate the Branch accounts.

Note that the relationship between data held in the Message Log and the Branch accounts is quite complex. Therefore a significant amount of knowledge and skill would be required to attempt this.

It should be noted that since R12 (July 2015) all access and actions carried out by Super Users to any database is strictly audited to an Oracle Audit table. The records in the Audit Table records the following information:

- User Id of the Super User
- Action (eg Log On, Execute a SQL command, Log Off etc)
- Date and Time of the action
- Actual SQL statement executed (where applicable)

This Audit Table is again extracted from BRDB soon after 1am and the data picked up and sealed before being copied to the Audit Server.

Note that it is possible for the Super User to manipulate the Audit table (including removing entries from the table. However should the table be removed entirely, then the database would stop working. If only old entries are removed, then the removel of entries will be recorded, thus making it clear that the table has been manipulated though the details of the changes may not be fully visible.

Prior to R12 (ie from 2010 to July 2015), only Log On and Log Off activities by Super Users were audited. For all legitimate access, then an MSC would have been signed off and the Logs of the Super User activities would have been attached to the MSC.

FUJ00087231

Therefore a correlation of Log On / Log Off activities of Super Users against MSCs should detect any rogue activities.

2.3 Detecting Changes to the Audit Trail

In order to make the changes to the Message Log described in section 2.2, the Super User would need Read access to the Key Store database which runs on the NPS and Read / Write access to the BRDB. Note that should the rouge application run on the BAL, then this isn't necessary as the BAL's have access to the Key store based on the IP address.

Note that the BAL Private Key only needs to be accessed once as the same key is used for a year.

However the BRDB would need to be accessed each day that the Message Log is to be corrupted.

All such access by a Super User would be audited in the Audit Table.

Therefore, should there be any allegations that any data has been corrupted, an examination of the Database Audit tables should ensure that this has not occurred. Although the Database Audit tables are not regularly examined they were recently checked as part of an external Audit of Horizon Online.

3. Old Horizon

On the old Horizon system, the mechanisms were very different from those used by Horizon Online. Section 3.1 provides an overview of how the Riposte Message store operates, then section 3.2 describes the Riposte Audit Trail. Finally section 3.3 shows how injections of messages by a super users would be detected in the audit trail.

3.1 Overview of Riposte

All Counter data was held in a bespoke Message Store which was part of the Riposte product supplied by Escher Inc. This data was replicated within each Branch to all counter positions and from each Branch to the Data Centres where it was held in the Correspondence Server Message Stores. Similarly, any data inserted into the Message Store at the Data Centre (eg Reference Data or authorisations for Banking Transactions) would be replicated back to the Branch Counters.

Selected data was then extracted from the Correspondence Servers to update Post Office Ltd's Back End systems.

All accounting at the counter was carried out based on the data held in the Message Store. The Riposte product managed the Message Store and it did not allow any message to be updated or deleted. Therefore all that could be done to corrupt the data in the Message Store was to inject additional messages which could then influence the Branch accounts. Such injections were possible at the Correspondence Server for users with sufficient access permissions.

Each message included 3 key bits of information which together provided a unique identification for each message:

FUJ00087231

- Group ID: this was the 6 digit FAD Code of the Branch with which the message was associated
- Node ID: This indicated the Counter Position at which the message was originally written for messages generated at the Counter or the Correspondence Server identifier for messages generated at the Data Centre. Counter Node Ids where between 1 and 31, and Correspondence Server Node Ids where between 32 and 63.
- Message ID: A unique number for each Group ID / Node ID. This number starts at 1 for the first message written at that Node, and increase by 1 for each subsequent message. This allows checks to be made that no messages are missing as that would result in gaps in the sequence of Message IDs

The concept of jsns used in Horizon Online was based on this.

Messages also have an associated "Expiry Date". This indicates the number of days after the message is first written before it can be deleted. An archive process ran on each counter and Correspondence Server at around 3am which deleted all messages that were past their Expiry Date, thus ensuring that the Message Store did not continue to grow indefinitely.

Some special messages which are referred to as "Persistent Objects" did not expire in this way but could be removed after they were replaced.

However again they were all held for a minimum of 34 days and in general were not relevant to generating the Branch Accounts.

In particular, Riposte was configured such that no messages where allowed to expire until they were at least 34 days old. This was to allow for counters that were offline for a significant period.

Each message also had an associated CRC, this was basically a checksum that was included to ensure that the message had not become accidently corrupted. Note that this was not a cryptographically secure seal and it would be possible for a sufficiently technically skilled person to alter a message and recalculate the CRC if they had access to the message outside the message store.

Due to the size of the Post Office Network, Branches were split into 4 separate Clusters. Each Cluster included 4 Correspondence Servers (2 in each Data Centre), thus ensuring that there were normally 4 copies of the data held in the Data Centres.

3.2 The Riposte Audit Trail

An Audit Application was run on the Correspondence Servers to take an audit copy of all data visible to that Correspondence Server.

The Audit Application was run on one Correspondence Server on each Cluster in each Data Centre. This means that there were two independent Audit Trails for each Branch. However when retrieving the data only one Audit Trail was used.

This application read every record that was visible to that Correspondence Server (ie all data in that Cluster) and wrote a text copy of that data to a text file. Each Audit application wrote data to 10 text files (based on one of the digits in the FAD Code)

FUJ00087231

and when the text file got to a certain size it was closed and a new file created for that text stream. The file included a hash value of the file contents to ensure that should it be accidentally corrupted, then this would be detected. Also around 1am each day the file was swapped thus ensuring that data associated with a given day was in discrete files.

Once these files had been written they became visible to the Audit server which would pick the files up and Seal them and store them until they are retrieved or deleted.

This process was not changed for Horizon Online.

Deletion of Audit records is currently suspended. They should be deleted after 7 years, but deletion was switched off sometime in 2014 (I think). Therefore all audit records since sometime in 2007 should be available. Those from before that time are no longer available.

If the audit trail is retrieved, then similar checks to those carried out on Horizon Online were made, namely:

- Each entire Audit File is checked to ensure that the digital seal stored at the time the Audit was produced (ie the day after the transactions took place) is valid.

Normally a Data retrieval will be for a number of days and so a number of Audit files will need to be retrieved. There would also normally be a number of audit files for each day.

- The data for the Branch in question is then filtered out from these audit files and checks are then carried out on a counter by counter basis as described below for the period of the extract:
    - No part of the Audit Data is missing or duplicated. This is done by ensuring that there are no missing or duplicate Message Ids for each counter / CS. The standard Audit Extracts into Excel include a report indicating that this check has been successfully carried out.
    - The CRC is recalculated and confirmed as correct for the message.

3.3 Detecting Changes to the Audit Trail

If a malicious Super User wished to interfere with the Branch's Data, then that would need to be done by injecting messages into the Correspondence Server or Counter Message stores using the Riposte APIs. Support Staff did have the capability (and occasional need) to do this at the Correspondence Server. Processes where in place to ensure that any such messages included information as to who had done this. Such information would not be visible in the standard audit extracts (but would be visible in a detailed examination of the raw audit data). Clearly any malicious corruption would be done without such trace information. However, if such data were injected at the Correspondence Server, it would be clear that this had occurred since the Node Id associated with the message would be that of the Correspondence Server at which the message had been injected and not a normal Counter Node Id. This would be clearly visible in any audit extract.

FUJ00087231

4. Postmaster's View

In the unlikely event that a Super User were to manipulate the transactions for a Branch, there is unlikely to be financial gain for the person doing the manipulation. Also any such changes ought to be immediately obvious to the postmaster since when they carry out their daily cash balances, they would find mismatches corresponding to the manipulated transactions.