INTERNAL AFFAIRS
Te Tari Taiwhenua

Te Puna Mātauranga o Aotearoa
NATIONAL LIBRARY OF NEW ZEALAND

Throughput efficiencies and misidentification risks in DROID version 6

This paper describes the technical testing of DROID v6, specifically the impact of using the new 'Max Byte Scan' function, and its impact on long term users of the DROID tool. The paper presents and discusses the findings of the tests, and concludes with some suggestions for future developments in both DROID and PRONOM that would result in significant efficiency gains by harnessing the new Max Byte Scanning function more completely in the PRONOM record.

"How does using the 'Max Byte Scan' function in DROID version 6 affect format identification?"

Jay Gattuso
Preservation, Research and Consultancy
National Library of New Zealand Te Puna Mātauranga o Aotearoa

Contents

Abstract

This paper describes the technical testing of DROID v6, specifically the impact of using the new 'Max Byte Scan' function, and its impact on long term users of the DROID tool. The paper presents and discusses the findings of the tests, and concludes with some suggestions for future developments in both DROID and PRONOM that would result in significant efficiency gains by harnessing the new Max Byte Scanning function more completely in the PRONOM record.

The Trigger

In November 2011, Ex Libris[1], the primary vendor for the National Library of New Zealand's (NLNZ[2]) digital preservation repository announced that changes would be made to some aspects of the repository (Rosetta[3]). These changes were expected as part of the usual software development that has seen Rosetta mature[4] as a fully functioning preservation-centric repository. This paper focuses on one of those changes, a proposal to integrate DROID[5] v6 into the file validation process. In previous versions of Rosetta, DROID versions 3 and 5 had been utilised, and this change to v6 was considered to be of significant interest to the Rosetta user community.

Addressing the Throughput Challenge

A significant challenge to collecting institutions that ingest large volumes of content into a single repository is the validation of those object at the time of ingest. In Rosetta, all ingested files are virus checked, and passed through DROID and other analysis/extraction tools such as JHOVE and NLNZ MDE. This process comes with a time / computational processing penalty, most notably in the virus checking and DROID aspect of the validation routine.

With DROID version 6 comes a new feature called 'Max Byte Scanning' (MBS) that attempts to significantly reduce the time taken by DROID to scan a single file.

The proposal by Ex Libris was to use the MBS function to in part address the throughput bottleneck, by applying it as a default setting inside the delivered version of DROID.

How MBS Works

When DROID scans a file, it takes the entire file, and scans the file looking for byte patterns that match with the registered signature. These signatures can be simple, and describe a short byte pattern at the beginning of file (BOF) or at the end of file (EOF). It can be a single byte long, or many bytes, including a variable offset that requires DROID to search over large portions of the file for a

[1] See http://www.exlibris.co.il/
[2] See http://www.natlib.govt.nz/
[3] See http://www.exlibrisgroup.com/category/RosettaOverview
[4] Project setup: 2004, Business as Usual System:2009
[5] See http://droid.sourceforge.net/

matching pattern. More details on this process can be found in the National Archive (UK) (TNA[6]) paper 'Digital Continuity DROID Guidance Version: 1.0'[7].

MBS works by essentially 'topping' and 'tailing' any file presented to it and constraining the scanned portion of the file. The area at the BOF and EOF that is scanned is set as a single variable in the options for DROID. The value set is 'Bytes' and can be any value from 1 Byte to ∞Bytes (where ∞Bytes is essentially a full file scan).

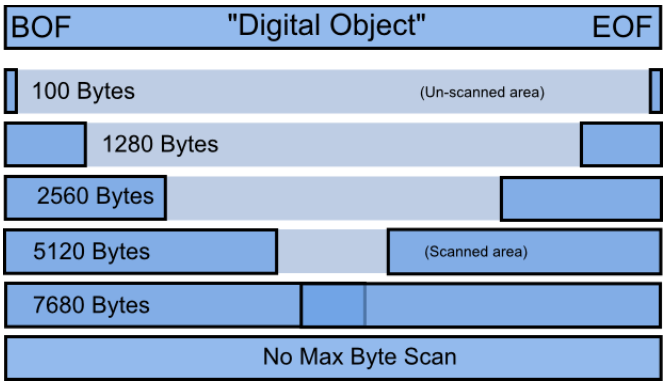

Figure 1 - Example of DROID MBS impact on a digital object

If the MBS value is smaller than half the file size, the middle 'section' is not scanned.

If the MBS value is larger than half the file size, but smaller than the whole file size, the middle 'overlapping section' is scanned twice.

If the MBS value is larger than the total file size, the whole file is scanned as if there was no MBS function applied.

A Mandate to Explore the Risks and Benefits?

One of the first questions we asked of this proposed development was:

> *"Is there a penalty for using this MBS function, on either accuracy of file identifications or on the number of formats that can be identified via DROID?"*

There appeared to be no definitive source of information that describes the various pros and cons of using the MBS. For NLNZ it was of particular interest as previous research work had identified some concerns with the inherent stability of the PRONOM signature/format description method and had noted some critical mutability in file format identifications over time as DROID signatures have matured[8, 9]. In an effort to protect the stability, and therefore long term value of DROID-based file

[6] See http://www.nationalarchives.gov.uk/
[7] See http://www.nationalarchives.gov.uk/documents/information-management/droid-how-to-use-it-and-interpret-results.pdf
[8] See http://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm

format identifications, NLNZ was concerned to know if using the MBS function would add another layer of mutability to the collective format identifications made via DROID by the Rosetta preservation repository.

Following some discussion with Ex Libris, the Rosetta user community and TNA, we decided to undertake some research to explore the impacts of deploying the MBS function in our live digital preservation system.

Setting up the Testing

To give some indication of the impact of the MBS function, a collection of files were taken from the preservation system and located on an external storage device. To ensure that only the byte signatures were tested, the files had their extensions stripped – preventing DROID from making any format identifications based on the DROID internal file-extension register.

These files were then checked with DROID version 6 (not using the MBS function) against format signature version 61[10] (with container signature v20120706) and any files that did not get a concrete match were removed from the test corpus.

This resulted in a test set of 11,177 files that spanned some 55 discrete formats (as identified by DROID v6 not using the MBS function).

These identifications were referenced as the 'ground-truth' for the testing.

Some of the format sets contained as many as 992 files and others only a single file. The constraining factor was simply the availability of files that were suitable for use in the test corpus, accessible to the tester, and of a type that DROID v6 could successfully identify (not using the MBS function) and using signature version 61.

Next, a series of DROID runs was completed with the MBS value set at a number of different values, ranging from 1 Byte to 8 Mbytes. The full list covered 41 MBS values:

[9] See http://www.nationalarchives.gov.uk/aboutapps/pronom/release-notes.xml
[10] See http://www.nationalarchives.gov.uk/documents/DROID_SignatureFile_V61.xml

Bytes	Kbytes	Mbytes
1	-	-
2	-	-
3	-	-
4	-	-
5	-	-
6	-	-
7	-	-
8	-	-
9	-	-
10	-	-
11	-	-
12	-	-
13	-	-
14	-	-
15	-	-
18	-	-
21	-	-
25	-	-
32	-	-
50	-	-
64	-	-
75	-	-
100	-	-
150	-	-
250	-	-
500	-	-
1024	1	-
2560	2.5	-
5120	5	-
7680	7.5	-
10240	10	-
20480	20	-
51200	50	-
65536	64	-
131072	128	-
262144	256	-
524288	512	-
1048576	1024	1
2097152	2048	2
4194304	4096	4
8388608	8192	8

Table 1 - MBS values used in testing

These values can be overlaid on the basic chart style used to document the results. (These can be found ….)

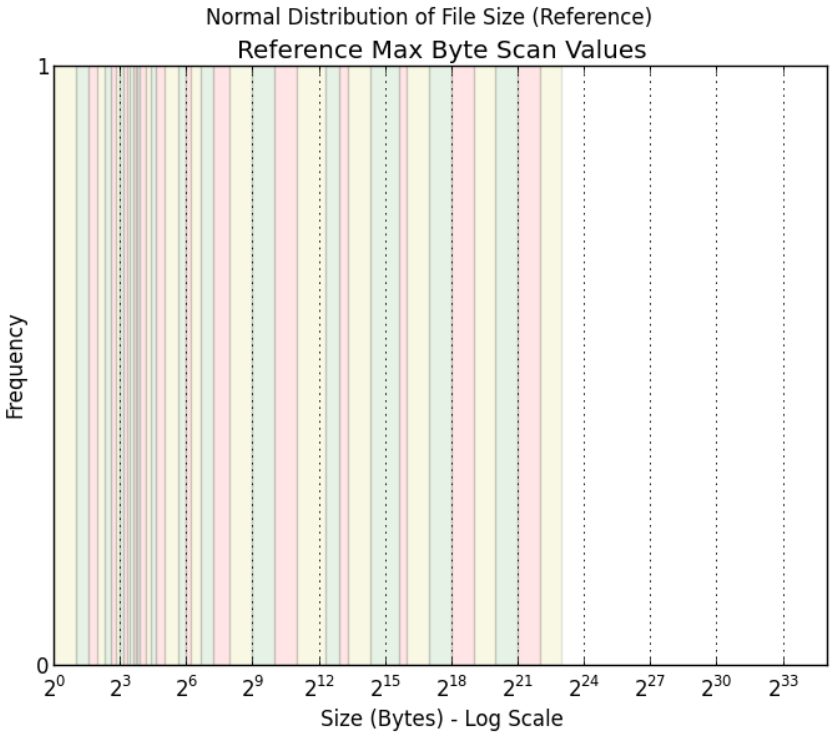

Figure 2- MBS Values over a log scale

The log files from each DROID run were exported from DROID, and ingested into a database. From there, they were queried, and the resulting data formed the main results for analysis.

The query broke down each of the log files into sets of results for each of the ground truthed reference PUIDs. It shows what file format IDs have been made by DROID during each max byte scanning value run, allowing the user to understand at what MBS value DROID settled on a single ‘correct’ PUID.

What was Discovered in the Testing?

A number of key findings were made as a result of these tests, the summary of which is reported as:

- 60% of the **files** tested had accurate format assertions with a MBS size of **≤64 Bytes**
    - This means that of the corpus of **11177 files**, more than half of the files tested could be accurately identified by DROID with a MBS size of **≤64 Bytes**
- 55% of the ground truthed **formats** were accurately identified with a MBS of **≤64 Bytes**
    - This means that of the set of **55 formats**, more than half of the formats tested could be accurately identified by DROID with a MBS size of **≤64 Bytes**
- 60% of the **files** tested never have an inaccurate format assertions made by DROID v6
    - This means that at all different MBS values, 60% of the 11177 **files** were only given either a '*No* ID' by DROID, or the correct 'ground-truth' PUID. No other PUIDs were offered at any time for these files
- 7% of the ground truthed **formats** required more than **64Kbytes** MBS to achieve an accurate format identification
    - This means that if the TNA recommended value of **65536 Bytes** (**64Kb**) is used, at least one file of 7% of the format sets tested do not on land the correct ground-truth format identity
- 30% of the **files** tested had an inaccurate format associated when MBS was used
    - This means that nearly a third of the tested files were given an incorrect format identity if an inappropriate (for the format in question) MBS value is used
- 9% of the ground truthed **formats** (representing 6% of the total tested files) resulted in an ambiguous format identification
    - This means that even in full scan mode, DROID is unable to determine a single format for these files, and can only offer a choice of between two and five, depending on the actual file in question

Results Summary Table

Definition of Terms Used

Ref PUID

This is the ground-truth format for a set of digital objects. This is the PUID assigned by DROID v6 in full scan mode.

Set Size

This is the number of files that comprise this set. This figure is included as it gives some indication to the degree of confidence that can be asserted against the set match.

Hard/Soft transition

This describes if the set undertook a "hard" transition, in that all the files flipped from No ID to the final PUID in one increment of the max byte scan value. If the set undertook a soft transition, the files flipped from '*No ID*' to the final PUID in more than one increment of the max byte scan value and/or via more than one PUID.

Start

This records the max byte scan value that at least one file from the set moved from '*No ID*' to any PUID. This figure is useful to understand when DROID starts to respond to the set.

End

This records the max byte scan value at which the last file from the set moved to the final PUID value. This figure is useful to understand when DROID settles on a PUID for the set.

Multi PUID per single item

This records if at any point more than one PUID is recorded for any single file in the set at any max byte scan value. This element is useful to see where there is an ambiguous match being made by DROID v6.

Drift

This describes any set that transitioned from No ID to its final PUID in a fragmented way over a number of max byte scan values. This element is useful to understand if the set/signature has a notion of 'framed' data inside the binary stream.[11]

Wrong PUID

This records if an incorrect PUID is recorded for any match at any max byte scan value. This element is useful to understand where DROID changes its decision with regard to the PUID it assigns to an object. An impact of this outcome is that incorrect use of a MBS value could result in wrong 'matches' being offered by DROID.

[11] An example of a framed data format is MP3. In a binary view of an MP3 file, data is segmented into 'chunks' or frames. These are used to present a decoder with a known block of data to work with, allowing some buffering to be undertaken in the decode method. The frame or block size is not stipulated in the standard, but is expected within a specific tolerance of sizes. Where DROID supports a variable pattern size, we expect that this is because the data format is expected to appear in identifiable blocks, but the block size is not fixed.

Ref PUID	Format Name	Set Size	Hard/Soft transition	Start	End	Multi PUID per single item	Drift	Wrong PUID
fmt-3	Graphics Interchange Format 1987a (gif)	6	H	5	5			
fmt-4	Graphics Interchange Format 1989a (gif)	19	H	5	5			
fmt-5	Audio/Video Interleaved Format (avi)	65	S	2560	20480		Y	
fmt-6	Waveform Audio (wav)	500	S	50	4194304	Y	Y	
fmt-11	Portable Network Graphics 1.0 (png)	17	H	15	15			
fmt-12	Portable Network Graphics 1.1 (png)	28	S	15	100		Y	Y
fmt-14	Acrobat PDF 1.0 - Portable Document Format (pdf)	1	H	7	7		N/A	
fmt-15	Acrobat PDF 1.1 - Portable Document Format (pdf)	43	H	7	7			
fmt-16	Acrobat PDF 1.2 - Portable Document Format (pdf)	509	H	7	7			
fmt-17	Acrobat PDF 1.3 - Portable Document Format (pdf)	518	H	7	7			
fmt-18	Acrobat PDF 1.4 - Portable Document Format (pdf)	615	H	7	7			
fmt-19	Acrobat PDF 1.5 - Portable Document Format (pdf)	652	H	7	7			
fmt-20	Acrobat PDF 1.6 - Portable Document Format (pdf)	634	H	7	7			
fmt-39	Microsoft Word for Windows Document (6.0/95) (doc)	11	S	7	150		Y	y
fmt-40	Microsoft Word for Windows Document (97-2003) (doc)	992	H	2	2			
fmt-41	Raw JPEG Stream (jpg)	500	H	2	2			
fmt-42	JPEG File Interchange Format 1.00 (jpg)	110	S	2	12			Y
fmt-43	JPEG File Interchange Format 1.01 (jpg)	500	S	2	12			Y
fmt-44	JPEG File Interchange Format 1.02 (jpg)	500	S	2	12			Y
fmt-49	Rich Text Format 1.4 (rtf)	39	H	3	3	Y		
fmt-50	Rich Text Format 1.5 (rtf)	68	S	3	18	Y		
fmt-53	Rich Text Format 1.8 (rtf)	93	S	3	51200	Y	Y	Y
fmt-61	Microsoft Excel Workbook (97-2000) (xls)	503	H	3	3			
fmt-96	Hypertext Markup Language (html)	3	S	8	500			
fmt-99	Hypertext Markup Language 4.0 (html)	25	S	25	32		Y	
fmt-100	Hypertext Markup Language 4.01 (html)	4	S	25	32		Y	
fmt-101	Extensible Markup Language 1.0 (xml)	500	H	13	13			
fmt-111	OLE2 Compound Document Format (office formats?)	6	H	7	7			
fmt-116	Windows Bitmap 3.0 (bmp)	155	H	9	9			
fmt-117	Windows Bitmap 3.0 NT (bmp)	1	H	50	50		N/A	
fmt-126	Microsoft Powerpoint Presentation 97-2002 (ppt)	32	H	7	7			
fmt-132	Windows Media Audio (wma)	49	S	15	7680		Y	Y
fmt-133	Windows Media Video (wmv)	1	S	15	10024		N/A	Y

Ref PUID	Format Name	Set Size	Hard/Soft transition	Start	End	Multi PUID per single item	Drift	Wrong PUID
fmt-134	MPEG 1/2 Audio Layer 3 (mp3)	488	S	32	2560		Y	
fmt-156	Tagged Image File Format for Internet Fax (TIFF-FX)	992	S	3	250		Y	Y
fmt-157	Acrobat PDF/X - Portable Document Format - Exchange 1a:2001 (pdf)	1	S	7	1048576		N/A	Y
fmt-198	MPEG Audio Stream, Layer II (mpa)	2	S	250	1024	Y	Y	Y
fmt-199	MPEG-4 Media File (mp4)	145	H	7	7			
fmt-214	Microsoft Excel for Windows 2007 (xlsx)	3	S	21	2560		Y	Y
fmt-276	Acrobat PDF 1.7 - Portable Document Format (pdf)	126	H	7	7			
fmt-279	Free Lossless Audio Codec (FLAC)	500	H	3	3			
fmt-353	Tagged Image File Format (tif)	8	H	3	3			
fmt-354	Acrobat PDF/A - Portable Document Format (pdf)	15	S	7	4194304		Y	Y
fmt-355	Rich Text Format 1.9 (rtf)	6	H	64	64			
fmt-412	Microsoft Word for Windows 2007 (docx)	10	S	21	1024		Y	Y
x-fmt-92	Adobe Photoshop (psd)	63	H	12	12			
x-fmt-135	Audio Interchange File Format (aiff)	1	H	3	3		N/A	
x-fmt-263	ZIP Format (zip)	16	H	21	21			
x-fmt-385	MPEG-1 Video Format (mpeg)	2	H	3	3			
x-fmt-390	Exchangeable Image File Format 2.1 (Compressed) (tif)	500	S	2	500		Y	Y
x-fmt-391	Exchangeable Image File Format 2.2 (Compressed) (tif)	500	S	2	500		Y	Y
x-fmt-394	WordPerfect for MS-DOS/Windows Document 5.1 (wps)	74	H	3	3			
x-fmt-398	Exchangeable Image File Format 2.0 (Compressed) (tif)	24	S	2	250			Y
x-fmt-409	MS-DOS Executable (exe)	1	H	1	1		N/A	
x-fmt-411	Windows Portable Executable (exe)	1	S	1	262144		N/A	

Total Files Scanned	11177
No. DROID Sessions	41
No. Format Sets	55
Total Number of DROID Assertions for all tests	458257

Table 2 - Main results Table for MBS testing

Another useful way of looking at the total corpus data is to look at the cumulative MBS values across all the format sets:

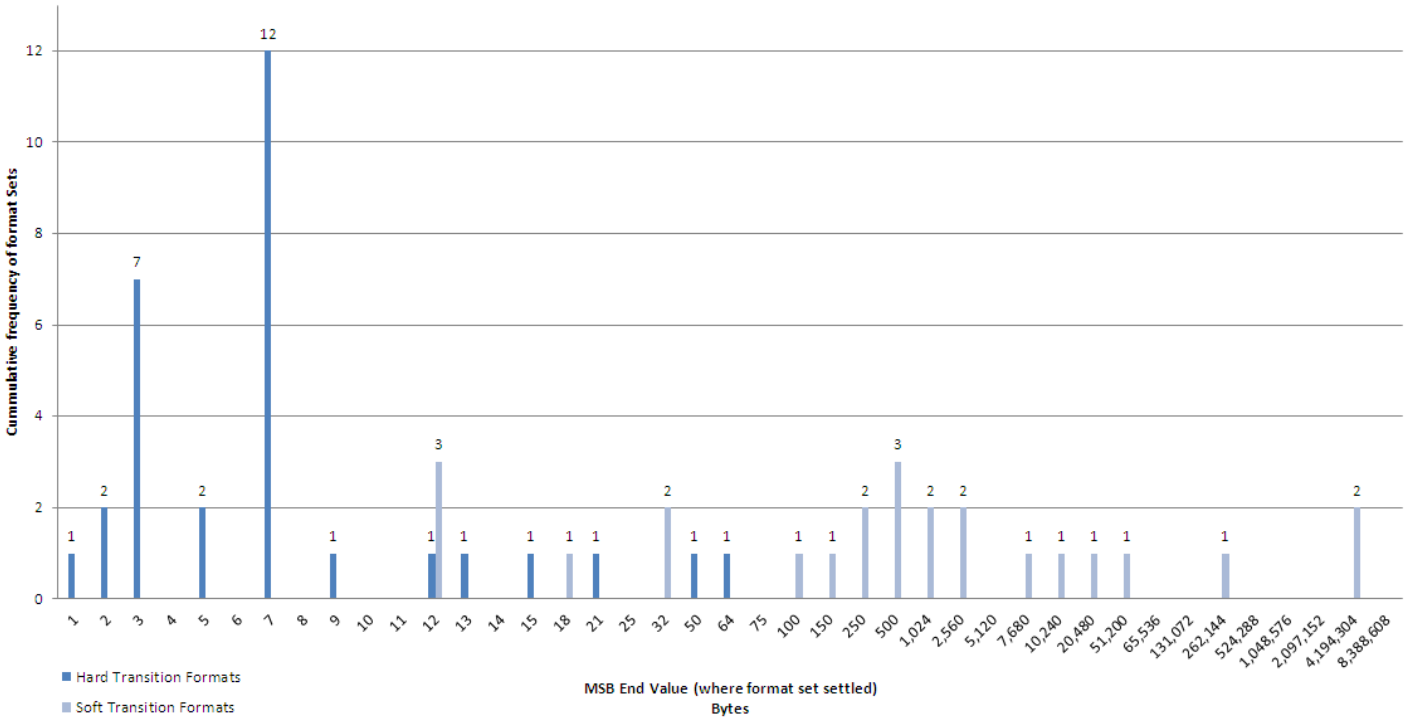

Figure 3 - Cumulative distribution of MBS End values

This demonstrates an interesting phenomenon. It is notable that all the 'Hard Transition' sets completed the single transition between 1 and 64 Bytes, conversely the 'Soft Transition' range from 12 bytes to 4 Mbytes as the lowest possible MBS value.

What does this all mean?

The most interesting (and valuable) finding is the potentially huge computational advantage to be made through the appropriate usage of the MBS function, especially where large files are being scanned.

While no specific individual timings were taken for the various DROID runs, it was notable that using the test corpus of 11,177 files, DROID took more than 5 hours in "full mode" to complete the run, and all the other test runs (40 different MBS values) could be completed while the full scan run was underway (with no more than 4 runs being active at any one time).

This anecdotally demonstrates the absolute throughput advantage that can be seen, especially for DROID/ Rosetta Users. This indicates that there is a real and tangible efficiency gain to be had, both on the time it takes for large digital objects to pass through DROID, and the amount of computational power required to complete an individual file scan by DROID.

At the lower end of the MBS range, having such a large number of positive and accurate matches completed with MBS values up to **64 Bytes** should be taken as an encouraging sign that there is some value in exploring this method further. To put this point another way, of the 11,177 files we tested, covering 55 different formats, 60% of files tested (made up of 55% of the formats we tested) achieved this swift and accurate result. There are clearly some gains to be had here.

At the higher end of the MBS range, having even a few formats that are not 'correctly' identified with a MBS size of **64 Kbytes** and higher should serve as a warning that without careful attention deployment of MBS can lead to inconsistent and erroneous DROID performance. That inconsistency can either come from use of different MBS sizes over time, or from trying to merge DROID results with and without the MBS function being used.

NLNZ has been using DROID as a part of its digital preservation repository for more than five years, and in this time we have collected a large number of records that link a digital object to an asserted file format. For reasons of consistency, it is essential that NLNZ has a high degree of confidence that any change to tools (or the deployment methods of those tools) does not change the format assertions that being made against an object, or types of objects over time in a damaging way.

Some change to DROID matches is inevitable as the PRONOM based signature knowledge base grows and matures. And as dedicated DROID users we endeavour to upgrade to the newest signature versions wherever possible (not least so we can utilise the signature patterns we have contributed to the PRONOM registry). With these new signatures often comes changes to existing signatures, and this is the change that concerns us.

Every care should be taken to lessen the impact of change on any tool users, and to minimise the level of inconsistency amongst tool matches made over time, and the level of analytical resource required to understand, document and promulgate the changes across legacy systems and records.

At a slightly higher abstraction, the results from these tests start to point towards an interesting but novel feature that could be included in DROID/PRONOM signatures. It's not difficult to envisage a new data element being added to the PRONOM record that describes the minimum MBS value that has been discovered for a specific format. This data element could potentially be used by DROID (or a pre-DROID file preparation process) to either:-

(a) present only the BOF and EOF parts of a file (based on the listed MBS, and where there is a high degree of confidence in the present file format e.g. if the file has come from a well-controlled object creation process such as an in-house digitisation program) to DROID for verification of file format

or

(b) to incrementally reduce the numbers of patterns being checked for by DROID when a file of an unknown format is presented to DROID. In this case perhaps DROID could be configured to start scanning with a MBS of 1 Byte, against all the known signature patterns, once completed it could drop any patterns that are known to trigger at 1 byte, and move to 2 bytes, scanning against the remain patterns. Once completed, any patterns that are known to trigger at 2 bytes are dropped from the scan and so forth.

Either of these approaches would deliver significant resource efficiency savings to processes that use DROID as a tool. The cost to implement this data element would be an extension of an element to the current PRONOM data model, and some community supported testing on the derivation of high confidence MBS values for PRONOM recorded formats.

In Summary

The final and underpinning question is one of absolutes. Much of the work undertaken in the format identification space leans on an implicit expectation that an absolute truth exists for format identifications and that at some point individual format signatures will resolve into a stable and immutable record.

Perhaps this expectation is flawed.

Perhaps it is only possible to have a relative 'truth' that must be asserted against a specific period of activity that is explicitly bound to a temporal paradigm. This paradigm is allowed to metamorphose as the core knowledge base expands.

If this is accepted as a premise, the next logical question to ask is what is the form of the connective tissue that links each generation of a signature set?

To be able to use these format-type assertions over time in a meaningful way there is an expectation of (1) a formal link between subsequent generations of assertions via signature changes and (2) a clear demarcation of boundary changes where any single descendant format signature traverses into a 'definition space' previously occupied by a predecessor.

If it were possible to:-

1) Design preservation systems that support a concept of a temporally asserted truth.

And

2) Design record structures and tools that support this notion of a temporal paradigm, and enable preservation systems to make asserted 'truths' at a critical point of enquiry in any digital objects lifecycle.

It is likely we would be building very different record keeping systems to those we see today. We would be unlikely to chisel individual matches into the 'stone' that forms the permanent AIP record, given the knowledge we have today of the mutability of that piece of information (and the subsequent impact of having to change that hard wired piece of data for many millions of digital objects)[12].

[12] The promise (and subsequent lure) of the semantic web maybe an almost self-selecting method of handling this mutability from an informational perspective, however it seems to be some time away from being an enterprise ready approach. In the mean time we have existing challenges to address using the tools available to us today.