Digital Data Preservation—A Viable Solution

Krishnan Arunkumar and Alagarsamy Devendran

Abstract Almost all the artifacts we have in digital only format is susceptible to loss because of the media deterioration, where they are stored. Even if we argue that migrating digital data on newer media at regular intervals will solve this issue. We have an even more important issue of digital data being inaccessible or not readable. This happens if software interpreting the data becomes obsolete. In such case the data is lost, as a bit stream is meaningless unless we can interpret them. Digital data preservation is a long and still open research area. There are various solutions proposed and implemented till date. All the solutions can be broadly classified into two categories (migration and emulation), based on the strategy to ensure longevity. We studied various approaches and strategies done till date for digital data preservation and propose a new framework—a combination of migration and emulation for digital preservation with fewer dependencies on future technology.

Keywords Digital data longevity · Digital data preservation · Data longevity

1 Introduction

In an interview with BBC at February 13, 2015—father of the Internet Vint Cerf, caused a stir when he expressed his concern that today's digital content might be lost forever. He also mentioned that if technology continues to outpace preservation tactics, future citizens could be locked out of accessing today's digital content enter a "digital dark age".

K. Arunkumar
ppltech, Chennai, Tamil Nadu, India
e-mail: emailarunkumar@gmail.com

A. Devendran (✉)
Dr. M.G.R. Educational and Research Institute, Chennai, Tamil Nadu, India
e-mail: devendran.alagarsamy@gmail.com

V. E. Balas et al. (eds.), *Data Management, Analytics and Innovation*,
Advances in Intelligent Systems and Computing 808,
https://doi.org/10.1007/978-981-13-1402-5_10

What is "digital dark age"?
It is the situation, where we have lot of digital data (documents and multimedia) but not able to read or interpret them. This could happen, if the format used to represent the digital data becomes obsolete.

2 Challenges in Digital Data Preservation

2.1 Core Problem

In digital preservation paradigm one of the most difficult challenges is to maintain the interpretability of data across changing technologies. A sequence of bits is meaningless if it cannot be decoded and transformed into some meaningful representation.

Example Documents stored as Word Star, a 1978 format, are as good as unreadable today.

2.2 Modern World Problem

Digital data exists as static content (like books, images, video, music) or dynamic content. In today's world most of the content in internet is rendered dynamically.

How do we preserver this type of content? Can we free these contents and preserver? How to exactly reproduce the dynamic content?

Dynamic content generation involves reproduction of the entire set of software executions. Preserving the entire set of "software execution" is a complex problem of aligning many moving parts over time. Over time lot of things are changing, hardware, OS, libraries, configurations, preferences, location and many.

2.3 Secured Documents Problem

Consider the case of encrypted documents and keys distribution for accessing them. Let us consider the case of storing encrypted documents in a distributed environment of many servers for long term. There is lot of research work happening in this area [1].

To achieve secrecy and longevity, we can have secret stored in multiple servers and have a constraint that a minimum number of shares among them must be required to reconstruct the secret. From a long-term archival point of view, any retrieval of such secured document needs some piece of software along with the data for construction of secret key from k or more pieces. Hence, software for key generation also needs to be preserved along with the platform on which it is developed.

2.4 Standards for Digital Data Preservation

Earlier research work on digital preservation was carried out by the Commission on Preservation and Access and the Research Libraries Group (RLG) in 1994. This task force summarized the basic requirements of preservation and issued a report (D. Waters and J. Garrett 1996). Between 1996 and 2002, many libraries, archives, and data-centers started to create digital repositories for their digitized information. The Consultative Committee for Space Data System (CCSDS) proposed an infrastructure model the Open Archive Information System (OAIS). The term OAIS also refers to the ISO OAIS Reference Model (ISO 14721:2003). This reference model is defined by recommendation CCSDS 650.0-B-1 of the CCSDS to standardize digital preservation practice and provide a set of recommendations for preservation program implementation.

3 Related Works

There has been significant amount of research work happened over the past 2 decades to address this problem. All of the long-term access strategies proposed can be broadly categorized under Migration and Emulation strategies.

3.1 Emulation

Here the system changes the environment, so that the original environment is created. Key thing is that the original digital item for preservation is intact and never modified. It is rendered by the application in the emulated environment.

Emulation has steadily improved over the last decade. At National Library of the Netherlands has used a UVC [2] (Universal Virtual Computer) system to make images available, and among other things via the EU project Planets developed the Dioscuri emulation system that can emulate x86 machine architecture. The project Preserving Virtual Worlds, which focuses on the preservation of games and interactive fiction, and the EU project KEEP (Keeping Emulation Environments Portable), also work with emulation. The status of this research is that there are promising prototypes, but products that can function in a preservation system are still far away.

Pros There are some types of records, for example interactive records, whose functionality cannot be preserved by the migration strategy. For such cases emulation is the only possible solution.

Cons Costs of such system are really high. Usually the following tasks are performed for data preservation:

- store bit streams
- preserve digital medium
- refresh/copy data to new media
- preserve integrity data during copy step
- preserve original application program, which was used to access data

Along with these steps for a proper working of emulation, we need to design and run emulator programs to mimic the behavior of old hardware platforms, OS and supporting libraries to run preserved application.

3.2 Systems Using Emulation Strategy

Dioscuri [3]—Dioscuri is x86 hardware emulator (written in java). It is OAIS complaint. It emulates various components of the hardware architecture as individual emulators and interconnecting them in order to create a full emulation process.

OLIVE [4] (Open Library of Images for Virtualized Execution)—Olive is a collaborative project for long-term preservation of software, games, and other executable content.

How they do it? Entire virtual machine along with the "executable content" is preserved along with metadata about the software and games preserved. VMs are stored with lot of metadata, so that users have lot of options to search and run. Virtual machine is then streamed using Internet Suspend/Resume technology. Users download the respective VM and run the preserved software and games from their local machine.

3.3 Limitations of Emulation Based Systems

Scalability
The ideas work well for artifacts from the pre-Web era. Considering the facts of huge number of web sites, the proposals of dedicated resources for the above methods need to be reconsidered.

Bit Preservation
Technical and Economic challenge of keeping the bits safe at huge scale cannot be simply ignored.

Intellectual Property
Digital copyright law on the data being preserved need poses lot of constraints to the emulation based solution.

Security Threats
Following are the major known threats on the virtualization technology.

- Hyper-jacking
- VM Escape
- VM Hopping

Technology Dependency
There are still a lot of dependency on the current technologies especially existence of JVM (Java Virtual Machine) in future.

3.4 Migration

Here the system changes the actual digital object, so that it adapt to the new environment to be migrated. This is akin to "recompilation". An earlier model could be portability through source code (for, e.g., a C program that is retargeted (re-compiled) from MIPS to x86-64); if source is not available but a binary is, then we may need to use the emulation method unless reverse assembly is feasible. However, the environment is typically complex and only part of this environment may be retargeted (firmware is almost always binary and proprietary).

This is an efficient approach as there is no interpretive overhead. But there is a startup cost and may be the "full system" needs to be converted. This latter aspect may be problematic if some migrated part is never used again.

Conversion to Standard Formats Another popular strategy is to migrate digital objects from many different formats to standard formats. For example, "Text documents" in several commonly available word processing formats to standards like SGML (ISO 8879). Images to standard file format and standard compression algorithms (e.g., JFIF/JPEG).

You can observe technical people used to migration of data from one data format to another format (especially documents). In the last decade this has become a primary interest to many long-term archival systems because of the less cost intensive task involved in this. There are various factors for this: Today fewer formats, each formats are used relatively longer and they are relatively better documented.

Definitely "Data" itself cannot be used. It requires a system that can interpret the data format in order to reproduce the data content. The more complex the data content and data format become, the greater the dependence on systems to interpret the data format. The migration strategy is thus by no means perfect, but currently in technical and financial terms the most simple and secure preservation strategy. Hence, used in practice by most preservation institutions.

Costs of Migration Apart from the regular bit stream protection, we have additional cost for building a monitor system which will

- Identify obsolete formats and of which the content must be migrated to other formats.
- Actual migration of data from one format to another.

3.5 Systems Using Migration Strategy

LOCKSS (Lots of Copies Keep Stuff Safe) [5] LOCKSS Program, from Stanford University Libraries and network, provides open-source digital preservation tools to preserve and authoritative access to digital content. It is OAIS complaint. It was started in 1999 @ Stanford University Libraries. Currently, there are 10,000+ e-journal titles from 520 publishers use LOCKSS. Libraries (United States and United Kingdom) that participate in the Global LOCKSS Network pay a small annual fee to join and get service.

Internet Archive Currently there are 361 billion web pages; 1,406,181 movies; 121,293 music concerts; 1,737,905 recordings; 5,279,432 texts are available.

Portico Porticos mainly involved in E-Journal Preservation Service. Currently there are 27,727,531 items available for preservation.

3.6 Limitations of Migration Based Systems

None of the above experiments really speaks about long-term preservation and also they heavily depend on current technologies. We are not very sure what future technologies will be? How, what and where information will be stored or processed? The argument within the community is like with continuous effort on migrating data at regular intervals will suffice.

Apart from the amount of effort in terms of finance and time it needs few more pressing disadvantages of migration strategy are

- Possible loss of information
- It requires manual/semi-manual checking of results
- Results are unpredictable
- It must be repeated every few years
- Error propagation

3.7 Insurable Storage Services [6]

This works deals with cost aspect for digital preservation and proposes an economically sustainable solution. The proposed solution creates a market place bringing two parties, people storing data for preservation and technology partners.

4 Our Contribution

We build a more concrete solution by a combination of migration, emulation, and encapsulation for digital preservation with less dependency. We looked at digital data being preserved and in retrieving the meaningful information from them in various scenarios and come up with a new concept of "Eternal Binary". A detailed discussion is given in the following section.

4.1 Digital Preservation Framework

Core Idea Packing a basic machine emulator (written in some abstract language), a simple Operating System to support the execution of Application along with the data to support the rendering of the same in future. Our claim is that in future technology, it will be easy to create an interpreter for this basic machine emulator packed along with data. With that interpreter we can emulate the basic machine, OS, Application and hence render data stored for longevity.

Eternal Binary With our system in place, every reader application (nothing but a native binary file specific to architecture) can be stripped and converted as "Eternal Binary" and from that moment all the data that are viewable via that reader can be viewed in future technology with the "Eternal Binary" being created through our system. This "Eternal Binary" application along with the "basic machine emulator", "simple OS" can render the data.
There are two ways to ensure the "Eternality" of the Application:

1. Create a distributed self-healing system to maintain all the "Eternal Binaries". So that any reference in future for viewing related data can be taken from it.
2. Always pack the "Reader Application" (Eternal Binary version), "basic machine emulator" and "simple OS" along with data. As of now, "Reader application developers" are responsible for creating the stripped version of the Application binary with given specification of basic machine and simple OS details.

In our experiments we created two such Applications, for reading and displaying ASCII encoded English txt file and "TSCII" encoded Tamil txt file. Later, we need an automated solution for converting any binary for a given architecture to an "Eternal Binary" file.

Even though our solution falls under "Emulation" category, our approach of stripping the minimal OS and basic machine emulation just to support the running of a single application only makes it unique and has the following key advantages:

Simplicity Proposed solution needs an interpreter, which can be easily implemented in future technology (compared to the complexity of hypervisors in virtualization technology.)

Security Proposed solution does not have the entire OS virtualized and hence exposing the security flaws of guest OS is avoided. Also, the code is open source which can be inspected for security loopholes.

Beyond 0s and 1s Proposed solution is not limited to a computers based on binary system (0s and 1s), even if the future technology has some other format of representing data. Creating an interpreter will be relatively easy with specification given.

5 Proposed Solution

5.1 Basic Machine Emulator—Von Neumann Architecture (MIPS R2000 ISA)

Almost all the computers we have today are based on the basic design "Von Neumann Architecture". As per "Von Neumann Architecture", all computers have

- Four main sub-systems: Memory, ALU, Control Unit, I/O
- Program resides in memory
- Program instructions are executed sequentially

Our emulator is built on Von Neumann Architecture. A detailed design/description for the same is out of scope of this document. Here, Execution of instruction happens in instruction cycle. This is the basic operation cycle of a computer.

5.2 Encapsulation/Packing

The digital data to be preserved (For our experiment we used data in "txt" files, both English language [ASCII] and Tamil language [TSCII]), basic machine emulator, simple operating system, application binary (simple "C" language program for reading and printing data stored in "txt" file). We believe carrying metadata about the requirement of interpreter and how to get display should be packed like a "readme.txt". This will help technologist of future to decode with their setup. This particular metadata is always kept updated (in latest standard format), may be every time a migration happens to avoid physical deterioration. A physical copy of the same is maintained with the digitally preserved artifact (Fig. 1).

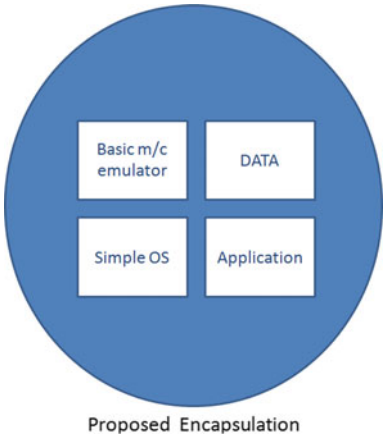

Fig. 1 Encapsulation and architectural stack

6 Experiments and Results

We created a POC (proof of concept) implemented in Java based on proposed framework for digital data stored in txt files. Hence, demonstrated the proposed framework will solve issues in digital data longevity. Following sections explains in details about the components of POC used in our experiments.

6.1 Interpreter (in Future Technology)

In our experiment we use applet-viewer, which can run in any standard browser. It will download the Basic machine emulator, OS, Application, data and render accordingly.

6.2 Base Machine Emulation

In our experiment we choose the following components:

Processor MIPS R2000 ISA implementation [7]

Co-Processor CP0, CP1 Interrupt/Exception handling and floating point operations

RAM Array of bytes (128 MB)

System Clock @ every nano-second (1 GHz Clock)

Once the base emulator is ready we hardcoded simpler programs like add, gcd for correctness of basic flow of instructions.

Another important aspect is the display module. Display is not a simple mapping of some region in memory to some patterns. It is not that simple. Basic VGA card and its programming are not trivial. We have used an open-source implementation (JPC Java X86 PC Emulator, which uses code from Bochs emulator) for our display system.

6.3 Binary of Application (as MIPS R2000 Machine Code)

For the test we needed a basic "hello world" program compiled for the MIPS R2000 architecture. We also need the program to be statically linked. We tried various cross compiler tool chains and got the basic program cross compiled for MIPS R3000 architecture with Linux. We ran the binary with QEMU-MIPS [8] (MIPS + Linux setup) and got required "Hello World" output.

6.4 Cross Compiler Tool Chains

With cross compiler—we can create executable code for any platform. For example: we can create executables for Linux/ARM on Linux/x86 based architecture.

For our experiments, we compiled for Linux/MIPS R3000 using "gcc binutils" tool chain. Details of how cross compiler works, the process to build a tool chain for a given target architecture and compile/build using the tool chain are out of scope of this document.

6.5 QEMU [8]

QEMU ("Quick EMUlator")—is an open-source processor emulator. It emulates CPU through dynamic binary translation and provides a set of device models, enabling it to run a variety of operating systems.

We used QEMU-MIPS to test our programs compiled for MIPS architecture.

6.6 ELF Files and ELF Loader

All the binary files generated by gcc and its tool chains are in ELF format. To run them we need a module to parse, interpret and load them in appropriate areas in memory. For that we implemented an ELF Loader parsing and loading corresponding sections into main memory.

6.7 System Call

A user program makes a "system call" to get the operating system to perform a service for it, like reading from a file, start another process, etc. In general system calls provides the essential interface between a user process and the operating system.

In our experiment, we ended-up in implementing the SYSCALL handler for 350 + system calls with very few actual implementations as per the trace from QEMU-MIPS trace.

6.8 Experiment Setup—Summary

Base Machine Emulator

- 140 Instructions (MIPS R2000)
- 128 MB RAM
- System Clock
- Co-Processors 0,1 Exception Handling and Floating Point operations
- 32 Integer Registers, Memory Address Register (MAR), Memory Data Register (MDR), Instruction Register (IR), Program counter (PC), HI, LO

Simple OS

- Interrupt handling and system calls implementation
- ELF Parser and Loader: To load and run single application
- File System (read from Actual data _le packed, write to Standard output device via VRAM)

Display

- VGA Card emulation
- Character mode initialization

Application

- C program with read, write to STD OUTPUT system call (statically linked and cross compiled for MIPS 3000 + Linux)

Data

- Simple txt file storing sentences in English encoded in ASCII format.

Package
All these information will be packed as a single JAR file and available for download/interpretation Interpreter. We depend on java's "applet-viewer" for downloading and interpreting the basic machine emulation, OS, Application along with data.

6.9 Results and Analysis

After the study of various contemporary approaches and strategies taken for digital preservation, we propose a new framework for digital preservation with very minimal expectation/assumption on future technology. We have come up with a proof of concept on how a data stored in our framework can be accessed in future after decades or even centuries. Of course, we assume the hardware availability and retrieval of bit streams is taken care and an applet-viewer like tool exists in future.

Digital data longevity is a deep recursive problem, which makes it difficult to come up with complete solution. We believe that packing entire computation system along with data and metadata for rendering is a reasonable solution. Our experiment on decoding the encapsulation with an applet-viewer, which is a part of every modern browser these days, proves our proposed framework works. Especially the second experiment we ran to retrieve non-Unicode TSCII encoded Tamil document. Retrieving such a document involves considerable time of either converting the document to Unicode document with some TSCII to Unicode conversion tool or by finding a font supporting TSCII encoding + installing it in the OS. If we cannot find them, then it is as good as lost data.

7 Conclusion

7.1 What Needs to Be Done Further?

Key differentiator of our proposal is combining Migration and Emulation strategy. We do a one-time migration to create an Eternal Binary out of application along with platform details. This step is to ensure the execution of application in future with very minimal dependency. That is, with the given data (0s 1s) stored privately/publicly and a reference to the Eternal binary stored in a common place. Retrieval of the data is guaranteed anytime anywhere in future.

After our experiments, we propose to build an open-source system like OLIVE to maintain all the Eternal binaries of possible applications generating digital data.

The system will have three main components

1. Interface for creating Eternal Binaries
2. Interface to link private/public data with available Eternal binaries
3. Interface to load and view data with Eternal binaries

7.2 Motivation

Scalability In the proposed solution, only the digital data needs to be preserved. This can be stored in a private or public cloud with different access privileges to ensure secured access over data. The application and platform to create or render the digital data are borrowed from the centralized repository, during the retrieval time. The complexity of the retrieval system is not a problem as we are going to maintain the eternal binary version of the same. With the implementation of bigger government projects like Aadhar, Digital India, e-Governance data preservation strategy ensuring security and scalability will be critical.

Intellectual Property As the data is separated and stored alone, every security feature available now can be applied.

Security Threats Proposed solution does not have the entire OS virtualized and hence exposing the security flaws of guest OS is avoided. Also, the code is open source which can be inspected for security loopholes.

References

1. Gupta, V. H., & Gopinath, K. (2006). An extended verifiable secret redistribution protocol for archival systems. In: *International Conference on Availability, Reliability and Security (ARES)*.
2. Van Diessen, R. J., & Lorie, R. A. (2005). IBM Research Report—UVC: A universal virtual computer for long-term preservation of digital information.
3. Rothenberg, J. (1999). Avoiding technological quicksand: Finding a viable technical foundation for digital preservation. A report to the Council on Library and Information Resources.
4. Gloriana, St. C., Linke, E., Satyanarayanan, M., & Bala, V. (2014). Collaborating with executable content across space and time. *ICST Transactions on Collaborative Computing*.
5. Dobson, C. (2003, February). From bright idea to beta test: The story of LOCKSS. Searcher 11.
6. Gopinath, K., & Simha, R. (2006). Insurable storage services: Creating a marketplace for long-term document archival. In *International Conference on Computational Science*, 3.
7. MIPS ISA. The MIPS R2000 Instruction Set manual. http://ti.ira.uka.de/TI-2/Mips/Befehlssatz.pdf.
8. Bellard, F. (2005). QEMU, a fast and portable dynamic translator. In *Proceedings of Usenix Annual Technical Conference*, Usenix Association.