When there’s a Linux in the PDF file

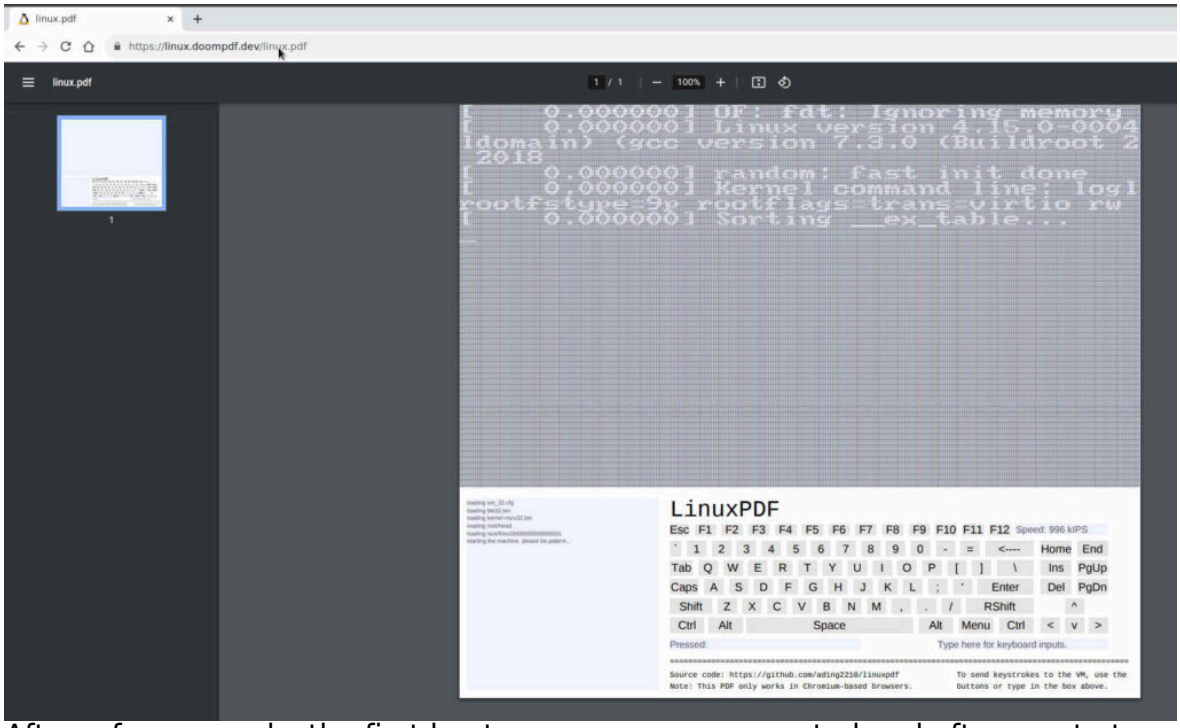
Egg, February 7, 2025: *The new mail archiving was introduced in the last blog with version 2025/II. Also new from version 2025/II are new rules on how documents are processed. Using the example of a PDF file that contains so much JavaScript code that an entire Linux is executed, we will illustrate why it makes sense to take great care when receiving documents — and why the ArchivistaBox is unique in this respect.*

Operating system runs in PDF file

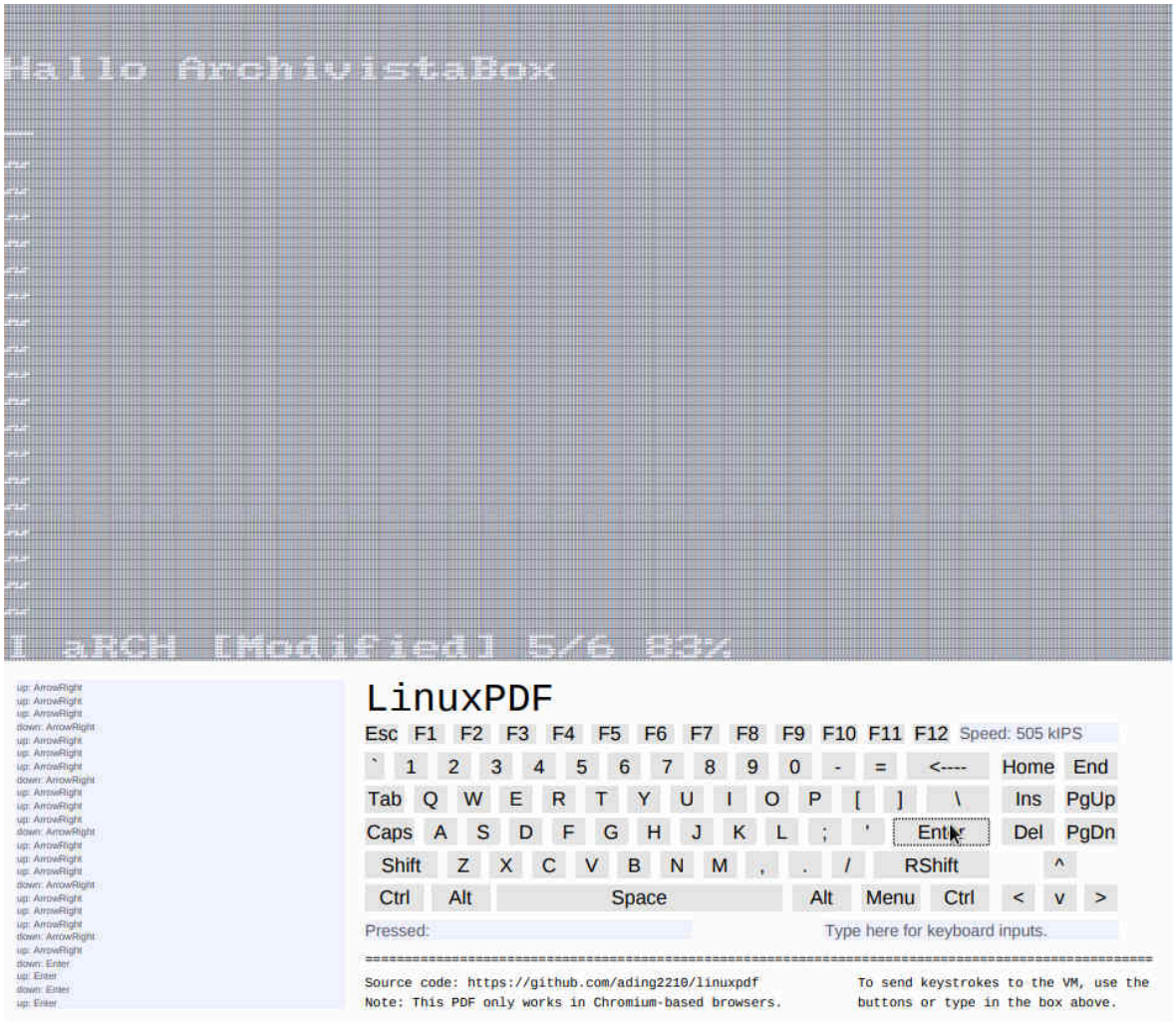
PDF files are generally regarded as the exchange format when it comes to exchanging content over the network. It can usually be assumed that these documents can be displayed on almost any platform (Windows, Mac, Linux, iOS and Android).

However, few people know that PDF files can also contain executable programs (keyword JavaScript). According to **an article on heise.de (German),** a resourceful tinkerer has put a complete mini-Linux into a PDF file. The **corresponding PDF file (approx. 5 MByte) can be downloaded from our server.**

A browser with a “Chrome engine” is required for the Mini-Linux to start; the JavaScript code does not work with Firefox or Safari. To start Linux in the PDF file, it is sufficient to start the linux.pdf file with Chrome (or Chromium, Edge not tested):



After a few seconds, the first boot messages are presented and after one to two minutes a Linux console is available. If you like, you can also use the text editor ‘vi’ to write a small greeting message:



It would be presumptuous to claim that Linux in the PDF file runs at high speed, the opposite is the case, it runs at a very leisurely pace. Therefore, the example can be considered a nice gimmick, can't it?

What is data, what is a program?

Let's summarize: a Linux is running in a PDF file of approx. 5 MB. Is it a file or is it a program or even an app? Will there soon be a PDF store? Or more generally, how can data be distinguished from programs?

Generally speaking, data is information that is not entirely volatile. It can be changed, but this happens from time to time, i.e. there are static states. With programs, the aim is to obtain (static) results after a greater or lesser number of actions, which are then (optionally) saved as data.

The programs themselves are usually a kind of black box. Even if the code is disclosed, only very few people are likely to know how programs work in detail. A small side note: Unfortunately, it is often the case that not even the programmers know later how their code works…

It is not difficult to see that there is no 100% demarcation, because a program is a static (executable) file. And because documents are mostly about the fact that they can be changed, it can make sense to integrate recurring processes for this data as macros or scripts in a file.

What would a homepage (html file) be these days without some snippets of JavaScript? From this perspective, it should be noted that JavaScript in PDF files does not necessarily have to be a problem.

If programs are started without being asked

However, the Linux PDF document shows the problem, because anyone who opens the file is not asked whether Linux should be started or not. The code is started automatically. Of course, it can again be argued that users are not asked whether JavaScript programs on websites should be started or not.

This may well be the case, but a homepage is not primarily a static file, unlike a PDF document, where information is generally exchanged in image or written form of a static nature. The problem with JavaScript in PDF files lies primarily in the fact that the code is started directly without interaction when the file is opened.

Security with the ArchivistaBox

PDF data is now part of the daily bread and butter of every ArchivistaBox and probably also of every document management system (DMS). With version 2025/II, the way in which documents are processed has been further refined. Previously, as soon as the ArchivistaBox was unable to assign a document to a suitable file format, PDF files were simply created from the original data using LibreOffice.

Incoming documents are now tested for executable programs. If this is the case, LibreOffice no longer attempts to prepare the data in a readable format by performing a text-to-PDF conversion.

This means that no virtual image of the corresponding files is created. And that brings us to the most important advantage that ArchivistaBox has had to offer for almost 30 years.

The “usual” case is not that programs are delivered, but that (non-exhaustive) documents, images, emails and/or multimedia data are managed. All incoming information is virtually “photographed”. This means that it is not necessary to open the data in ArchivistaDMS for later viewing.

Instead, the created image data is displayed. This means that it is not necessary to start any scripts or programs, e.g. to display PDF and/or Office documents via a viewer, as is the case with (all) other solutions. This means that ArchivistaDMS is not “vulnerable” to scripts or programs. Where no programs are started, there is no problem with security.

This means that our customers alone decide which tools are used for their data. At Archivista GmbH, for example, we work 100% without Microsoft Office packages, and there is also no Outlook or Teams. LibreOffice and the PDF viewer are configured so that no programs are started when files are opened.

Practical tip: All those who work with Windows or Mac can deactivate the automatic start of scripts in files in the Office programs and also in the PDF viewer. Even better (if possible) is to disable the launching of programs in documents altogether.

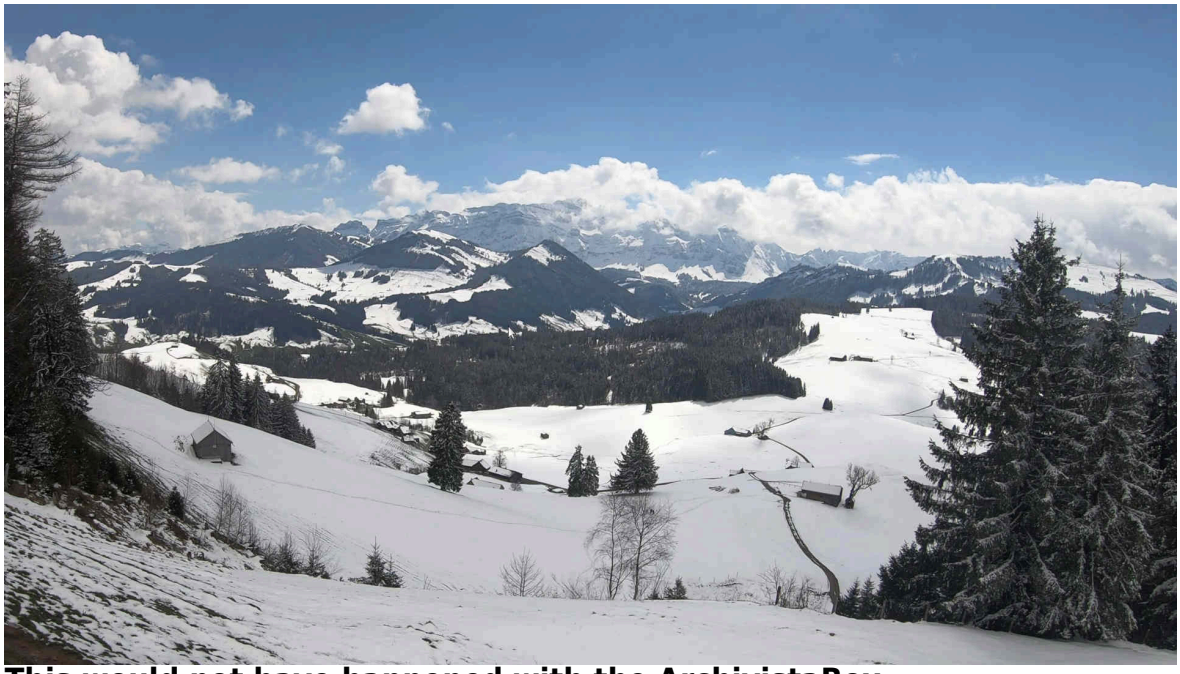
Maximum minimized risks

As long as no code contained in documents is executed when opening or processing documents (and this is 100% the case with the ArchivistaBox), there is in fact no corresponding risk (security by design).

Equally unique is the fact that the ArchivistaBox works in the main memory. Even if the customer (e.g. via the ArchivistaBox desktop) should install additional programs (with malicious code) himself, the malicious code can be removed again by restarting the ArchivistaBox, as the ArchivistaBox is completely (and automatically) reinstalled in the main memory each time it is restarted.

The problem remains that operating systems can also contain malicious code. With open source, however, any vulnerabilities can be rectified very quickly because the source code is known. If necessary, all customers receive the latest versions within hours. The update process is always started from the ArchivistaBox (WebConfig); there is no automated access from the supplier. This is also a plus in terms of security, as it prevents automated attacks.

Currently, the vast majority of IT solutions no longer work without a connection to the Internet. The ArchivistaBox can easily be operated without a network, with a mouse, keyboard and screen. This may seem exaggerated by today’s standards, but there are still good reasons for this scenario. One example is self-supporting archives. Documents are exported to a separate ArchivistaBox on the basis of a query. This can then be started in a “cage” (own computer or with virtualization) without having to establish a connection to the network.



This would not have happened with the ArchivistaBox
As a conclusion, anyone who thinks that attacks via PDF files (or documents in general) are not a problem or would not take place is welcome to visit the Whatsapp worm page (I know, it's tabloid, but written in an understandable way), where the devices were successfully attacked via prepared PDF files (analogous to linux.pdf).
Once again, attacks launched in this way are not possible via the ArchivistaBox because, firstly, image data is created from the PDF files and the JavaScript fragments are not processed. And secondly, the image data is displayed in ArchivistaDMS instead of the PDF files. In this sense, the ArchivistaBox is ideally equipped for tough day-to-day business.
This does not mean that no further improvements are possible. With version 2025/II in particular, the new document parser ensures that problematic documents are no longer processed at all. With this in mind, we recommend that all customers upgrade to version 2025/II.
However, the new security concept of the ArchivistaBox 2025/II is only effective if the updates are also installed promptly. Call up WebConfig and start the online update, that's all it takes. Optionally, we can update the ArchivistaBox for our customers at regular intervals, because the best security concept is of little use if the corresponding adjustments exist but have never been installed.

Facebook
Twitter