osFree demo CD (some technical details)

We have uploaded our new osFree 0.0.4.1 ISO image Since the previous osFree ISO image 0.0.4, it has some advancements. For example, the previous setup used a small 500-byte executable, which used only one OS/2 API function – DosPutMessage, imported from a single msg.dll (which is a forwarder to doscalls.dll), the new setup includes a bigger minicmd.exe LX executable (9 kilobyte in size), which uses more OS/2 API's.

It must be noted, that these executables still have no thunks for calling 16-bit API's. For that, we use our own invention, sub32.dll, which stands for “VIO/KBD/MOU 32-bit Subsystems”. This DLL is treated a special way by our LX loader – an executable is linked against EMXWRAP.DLL, which is a 32-bit wrapper over VIO/KBD/MOU 16-bit API's, taken from EMX. This way gcc compiler and other GNU packages can work with only 32-bit API's, and do not use 16-bit ones. The same way, we use 32-bit API's only. But when our LX loader sees the reference to EMXWRAP module, it resolves this reference by linking with SUB32. So, SUB32.DLL is a kind of EMXWRAP alias. Our L4 implementation of OS/2 API, in our demo, runs a mincmd.exe test executable, linked with SUB32 and DOSCALLS, whereas on IBM's OS/2, it depends on EMXWRAP.DLL from EMX runtime. But when EMX EMXWRAP.DLL wraps 16-bit API's by 32-bit ones, osFree SUB32 is a real 32-bit implementation, which does not depend on 16-bit API's.

Similar 32-bit VIO/KBD/MOU API's had the IBM's OS/2 Warp Connect (PowerPC Edition). So, it is not new. In the future, we plan to add support for 16-bit OS/2 API's for binary compatibility. It is planned to make an implementation of LX loader which will on-the-fly convert a mixed 16/32 bit executable to a pure 32-bit, changing all 32→16 bit thunks on the fly, to the calls to pure 32-bit API's.

DOSCALLS.DLL is a virtual DLL in a present OS/2. It imports API's from oS/2 kernel. There is DOSCALL1.DLL too, it is an ordinary disk file, whereas DOSCALLS.DLL is a virtual module, which does not exist on the disk, and it references API's, present directly in OS/2 kernel, or indirectly, through functions on doscall1.dll, which in turn, call OS/2 kernel functions.

In osFree, there is no DOSCALL1.DLL, but DOSCALLS.DLL is present as a disk file, not as a virtual module. In our design, there exist a virtual kal.so module, which calls RPC's to OS/2 server. DOSCALLS.DLL is a real disk file, which exports all OS/2 API's, and wrappers over kal.so functions.