PE应用跨平台跨架构移植实践

江苏润和软件股份有限公司　魏建刚/Wei Jiangang

开放原子开源基金会 OPENATOM FOUNDATION | OpenEuler

目录

PE应用跨平台移植的背景与问题

需求：

同架构、不同操作系统平台间移植执行；
不同架构+不同操作系统平台间移植执行；

此外，还需关注32位与64位差异性；

```
root@wei-VirtualBox:~# cat Hello.c
#include <stdio.h>

int main()
{
    printf("Hello World!\n");
    return 0;
}
```

objdump -d -M intel Hello-win.exe

```
……
0000000000401640 <__main>:
 401640:    8b 05 ea 59 00 00      mov    eax,DWORD PTR
[rip+0x59ea]       # 407030 <initialized>
 401646:    85 c0                  test   eax,eax
 401648:    74 06                  je     401650 <__main+0x10>
 40164a:    c3                     ret
 40164b:    0f 1f 44 00 00         nop    DWORD PTR
[rax+rax*1+0x0]
 401650:    c7 05 d6 59 00 00 01   mov    DWORD PTR
[rip+0x59d6],0x1       # 407030 <initialized>
 401657:    00 00 00
 40165a:    e9 71 ff ff ff         jmp    4015d0
<__do_global_ctors>
 40165f:    90                     nop  ……
```

objdump -x Hello-win.exe

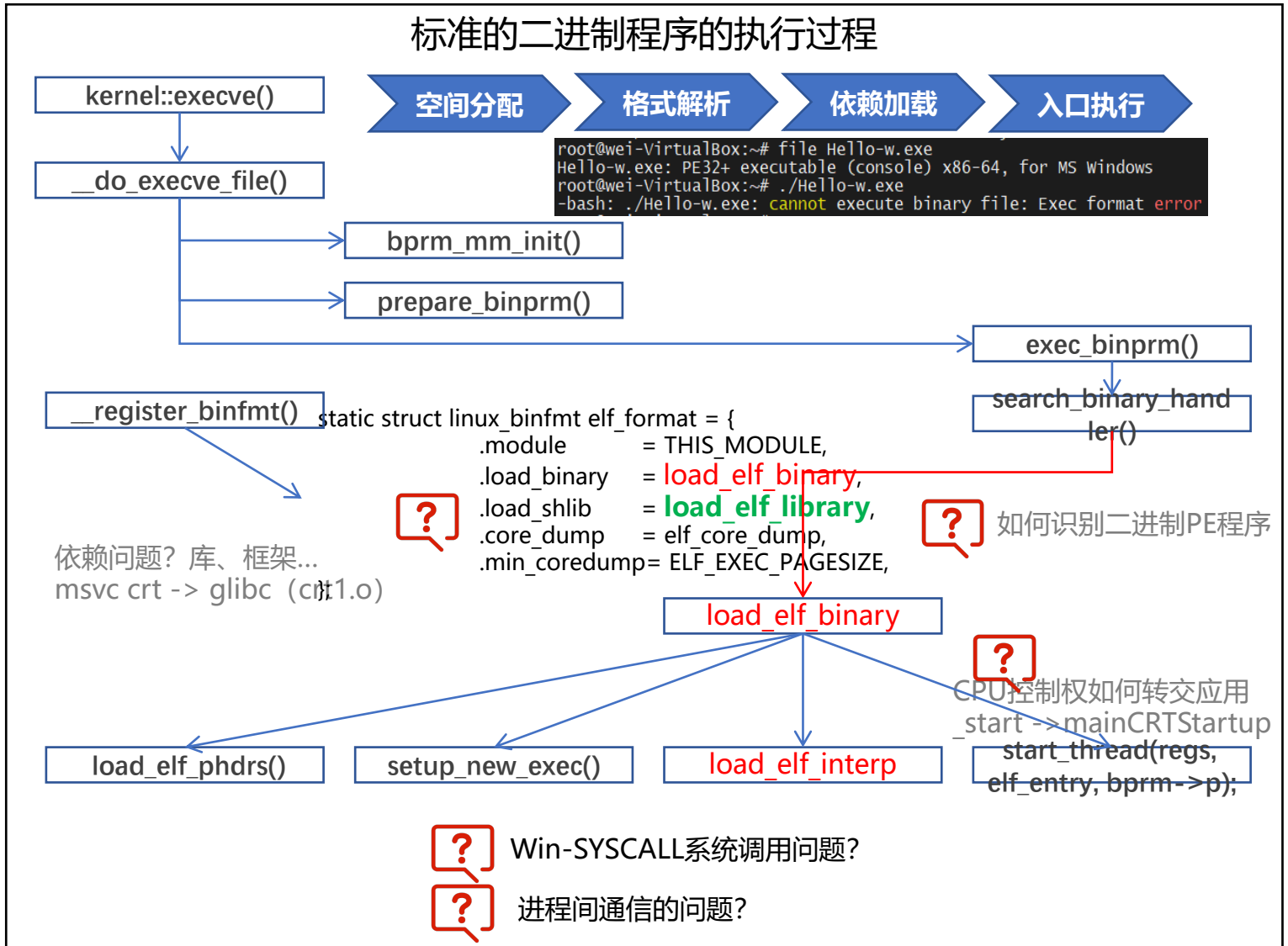
```
……
Sections:
Idx Name          Size      VMA               LMA
File off  Algn
  0 .text         00001d18  0000000000401000
0000000000401000  00000600  2**4
                  CONTENTS, ALLOC, LOAD,
READONLY, CODE, DATA
  1 .data         000000c0  0000000000403000
0000000000403000  00002400  2**4
                  CONTENTS, ALLOC, LOAD, DATA
  2 .rdata        000008a0  0000000000404000
0000000000404000  00002600  2**5
                  CONTENTS, ALLOC, LOAD,
READONLY, DATA
  3 .pdata        00000258  0000000000405000
0000000000405000  00003000  2**2

                  CONTENTS, ALLOC, LOAD,
READONLY, DATA
  4 .xdata        000001e8  0000000000406000
0000000000406000  00003400  2**2
                  CONTENTS, ALLOC, LOAD,
READONLY, DATA
  5 .bss          00000980  0000000000407000
0000000000407000  00000000  2**5
  ……
```

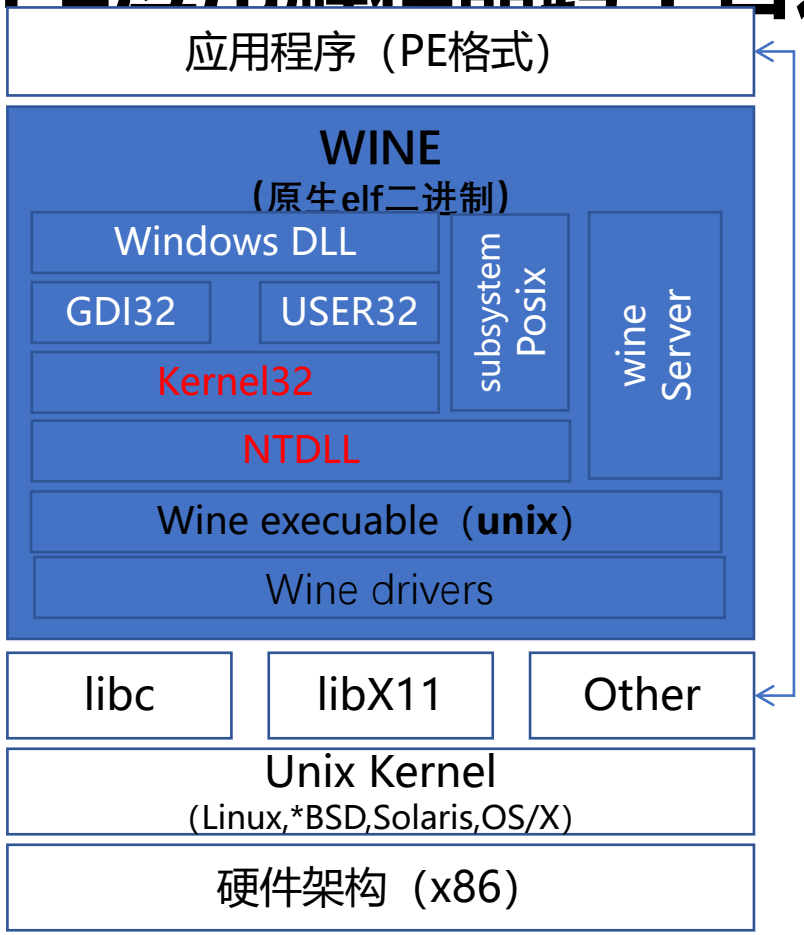
总结：从文件结构（.text）与序列化的CPU指令可知，PE文件内的代码在Windows和Linux之间具备"可移植 "的条件（CPU架构相同）！

标准ELF格式应用运行过程

```
root@wei-VirtualBox:~# file Hello
Hello: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=dc35fc3d1674f1a20f64a3652aadbbc6054d250f,
for GNU/Linux 3.2.0, not stripped
root@wei-VirtualBox:~# ldd Hello
        linux-vdso.so.1 (0x00007fff903f5000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6
(0x00007f477275c000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f4772972000)
root@wei-VirtualBox:~# strace ./Hello
execve("./Hello", ["./Hello"], 0x7ffca18af4b0 /* 34 vars */) =
0
brk(NULL)                               = 0x5633d4342000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffcf0667910) = -1
EINVAL (Invalid argument)
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No
such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache",
O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=117205, ...}) = 0
mmap(NULL, 117205, PROT_READ, MAP_PRIVATE, 3, 0) =
0x7f668de28000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6",
O_RDONLY|O_CLOEXEC) = 3
……
fstat(1, {st_mode=S_IFCHR|0600, st_rdev=makedev(0x88,
0x2), ...}) = 0
brk(NULL)                               = 0x5633d4342000
brk(0x5633d4363000)                     = 0x5633d4363000
write(1, "Hello World!\n", 13Hello World!
)         = 13
exit_group(0)                           = ?
+++ exited with 0 +++
```



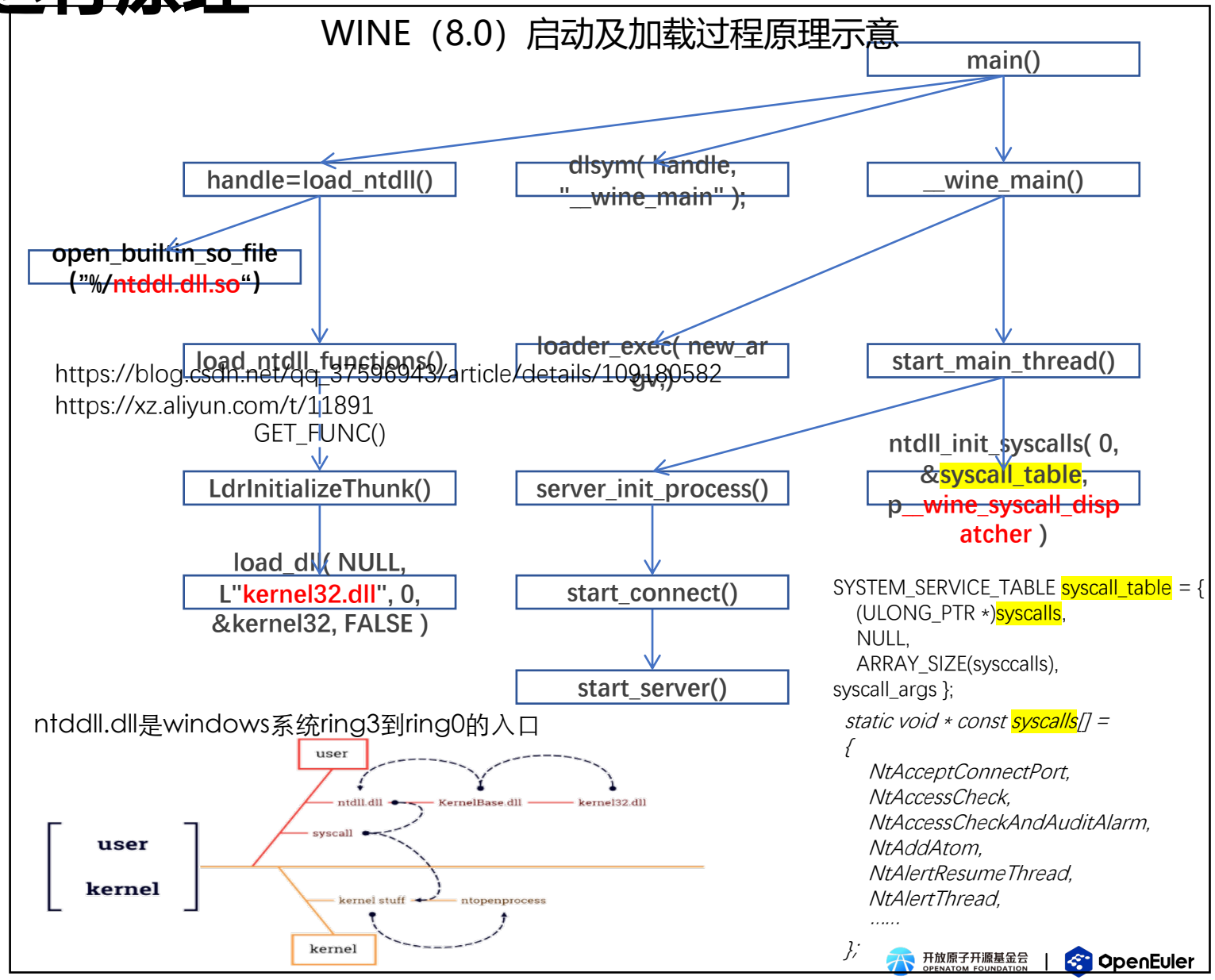
PE应用解释器跨平台运行原理



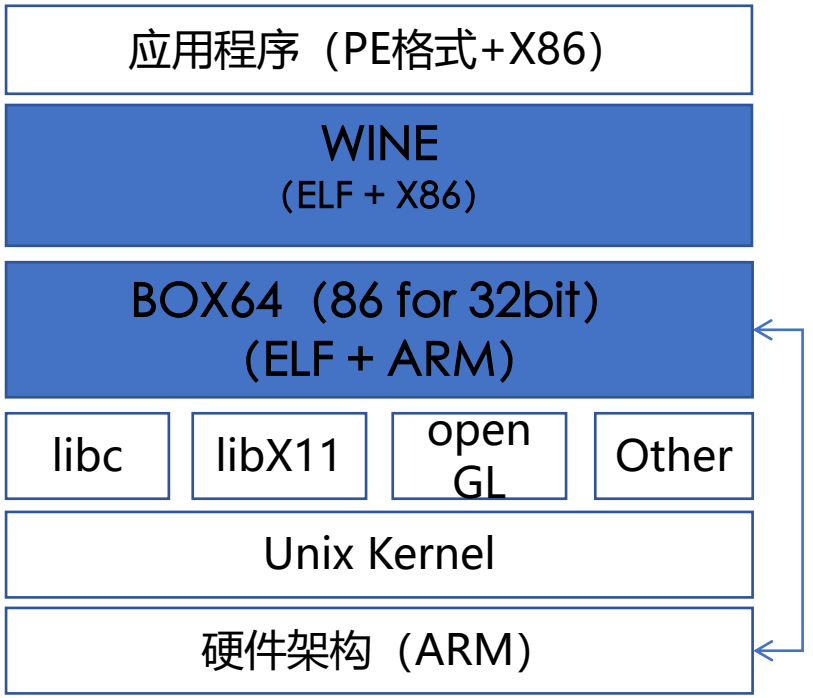
说明：

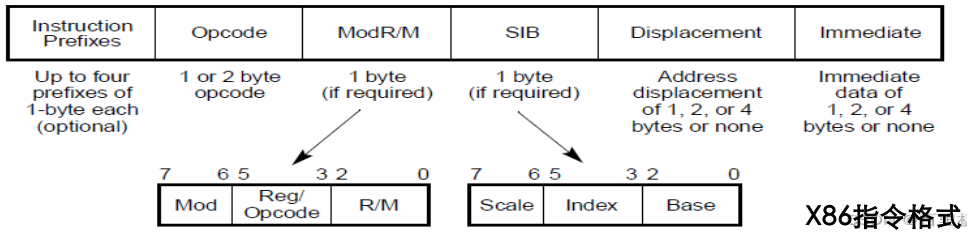
I. wine主要通过实现Windows API来模拟Windows操作系统的行为; 运行时拦截系统调用，并转换为对目标系统的调用。

II. wine server提供进程间通信（IPC）、同步和进程/线程管理。

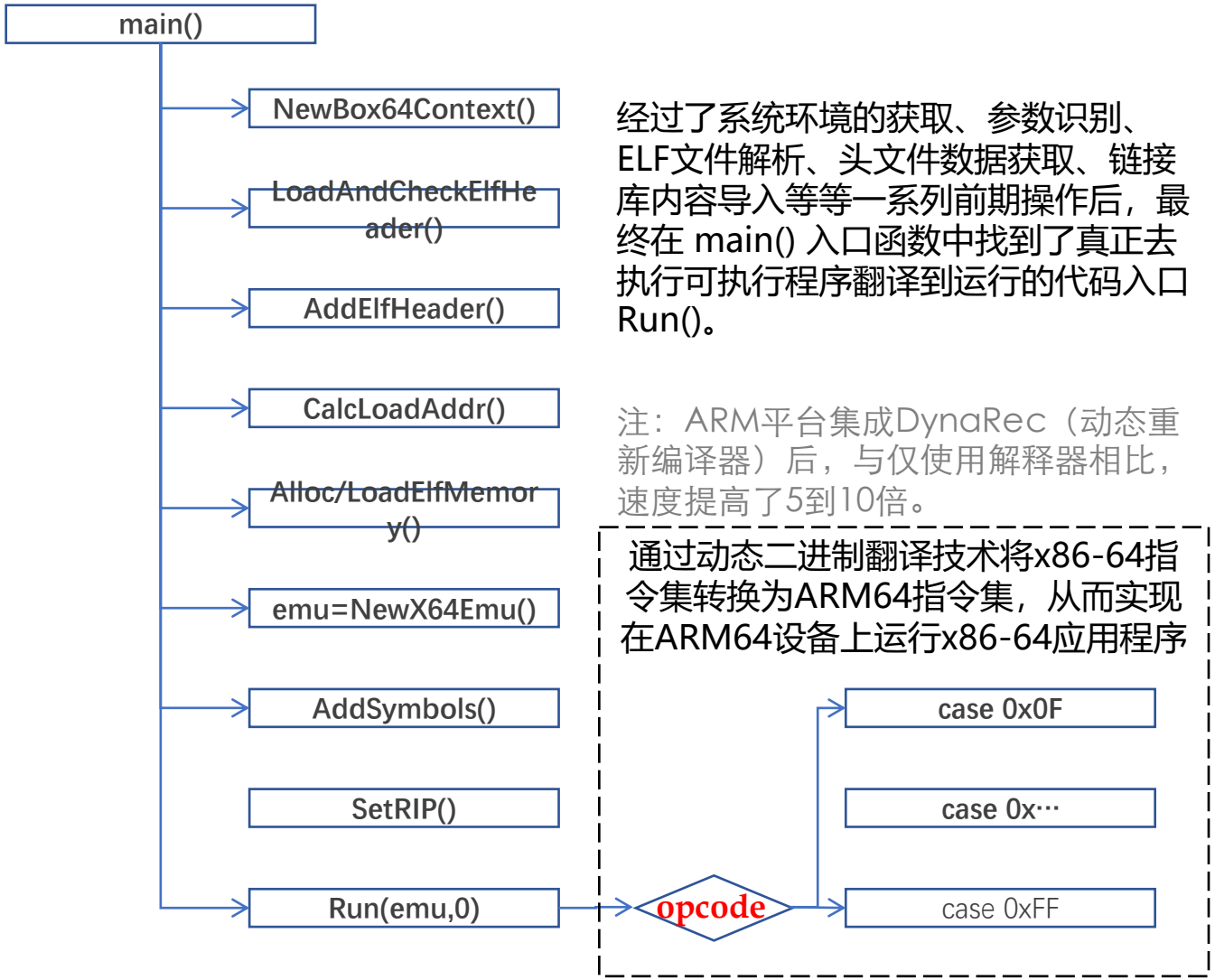


PE应用解释器跨架构运行原理（1）





X86指令格式

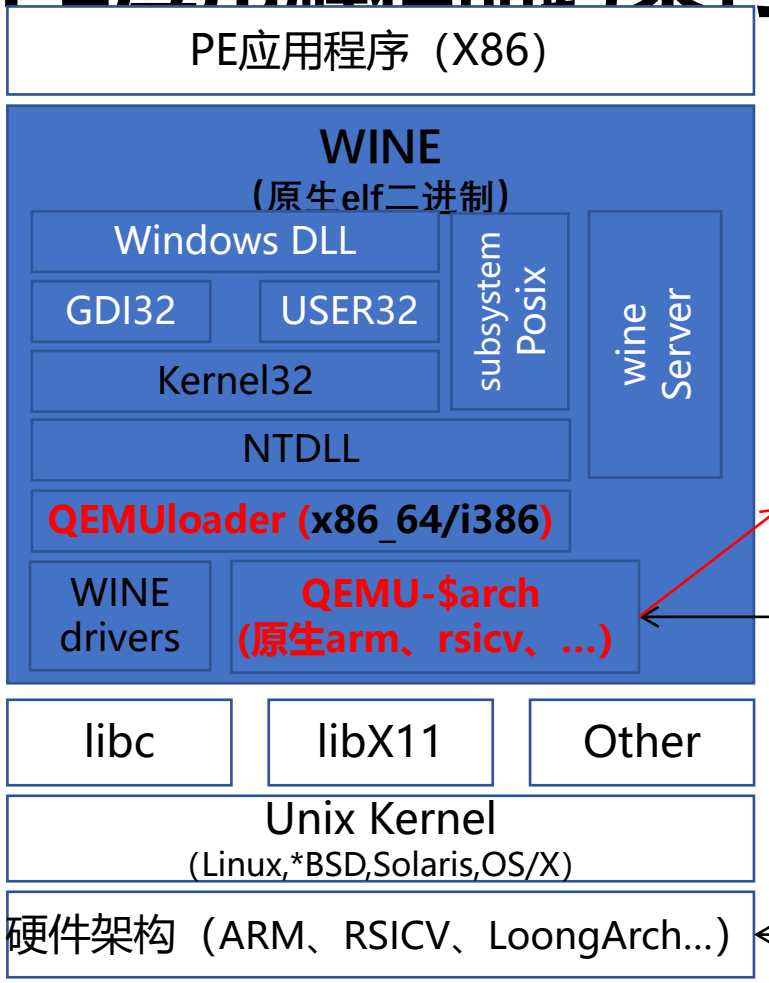


经过了系统环境的获取、参数识别、ELF文件解析、头文件数据获取、链接库内容导入等等一系列前期操作后，最终在 main() 入口函数中找到了真正去执行可执行程序翻译到运行的代码入口Run()。

注：ARM平台集成DynaRec（动态重新编译器）后，与仅使用解释器相比，速度提高了5到10倍。

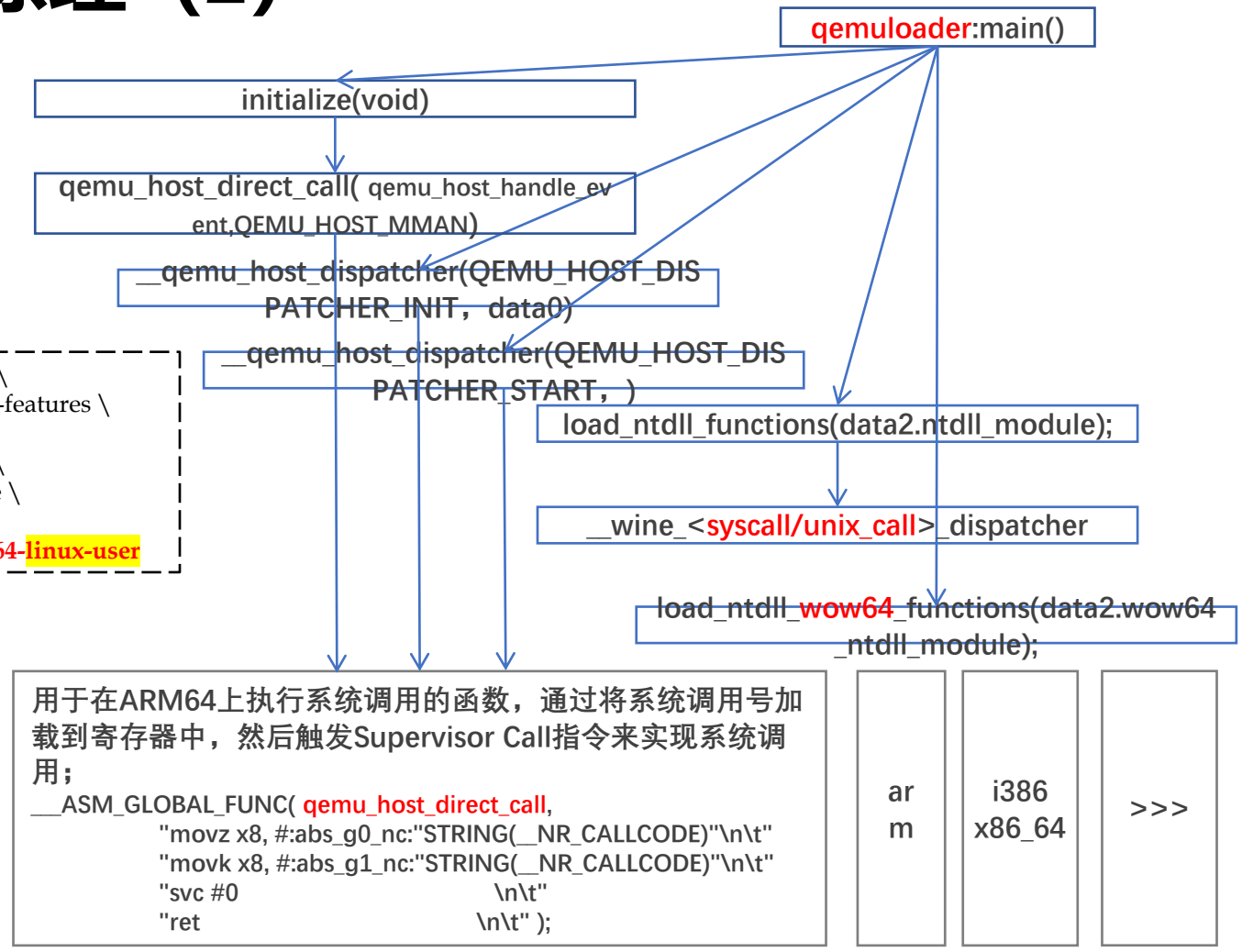
通过动态二进制翻译技术将x86-64指令集转换为ARM64指令集，从而实现在ARM64设备上运行x86-64应用程序

PE应用解释器跨架构运行原理（2）




```
qemu/configure \
--without-default-features \
--disable-fdt \
--disable-system \
--enable-ca11c0de \
--disable-rcu \
--target-list=x86_64-linux-user
```

动态二进制指令翻译 与 Unix库调用转发



用于在ARM64上执行系统调用的函数，通过将系统调用号加载到寄存器中，然后触发Supervisor Call指令来实现系统调用；

```
__ASM_GLOBAL_FUNC( qemu_host_direct_call,
                   "movz x8, #:abs_g0_nc:"STRING(__NR_CALLCODE)"\n\t"
                   "movk x8, #:abs_g1_nc:"STRING(__NR_CALLCODE)"\n\t"
                   "svc #0                                      \n\t"
                   "ret                                         \n\t" );
```

arm | i386 x86_64 | >>>

Wine-CE 仅模拟基于客端指令架构的Windows动态链接库和qemuloader，并将针对Unix库的调用转发到主端执行。基于非必要不模拟的原则，使它拥有较好的性能。

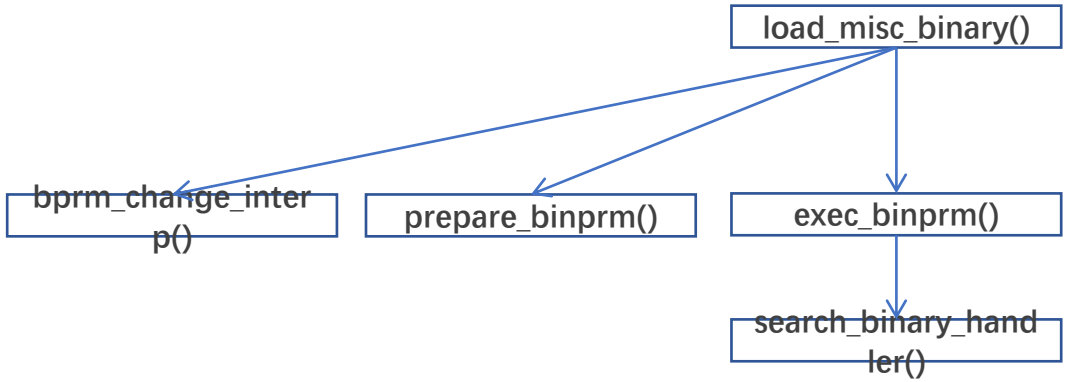
项目地址 https://gitee.com/wine-ce/wine-ce

```
export LD_PRELOAD=$wine_path/ntdll.so:$wine_path/win32u.so;
exec qemu-x86_64 qemuloader  windows-app.exe
```

Miscellaneous Binary Format 机制

```
root@wei-VirtualBox:~# file ./Hello-w.exe
./Hello-w.exe: PE32+ executable (console) x86-64, for MS Windows
root@wei-VirtualBox:~# objdump -p ./Hello-w.exe | grep "DLL"
 vma:            Hint    Time      Forward  DLL       First
        DLL Name: KERNEL32.dll
        DLL Name: msvcrt.dll
root@wei-VirtualBox:~# ./Hello-w.exe
-bash: ./Hello-w.exe: cannot execute binary file: Exec format error
root@wei-VirtualBox:~# strace ./Hello-w.exe
execve("./Hello-w.exe", ["./Hello-w.exe"], 0x7fff0ca4fe60 /* 34 vars
*/) = -1 ENOEXEC (Exec format error)
strace: exec: Exec format error
+++ exited with 1 +++
```

```
static struct linux_binfmt misc_format = {
        .module = THIS_MODULE,
        .load_binary = load_misc_binary,
};
```



:name	:type	:offset	:magic	:mask	:interpreter	:flag

- :interpreter → 启动文件的程序（绝对路径）
- :magic → 不仅可以通过文件的扩展名来判断，还可以通过文件开始位置的特殊的字节（Magic Byte）来判断
- :type → 表示如何匹配被打开的文件，取值为 E 或 M 。E 表示根据扩展名识别， M 表示根据文件Magic Bytes来识别
- :name → 规则名

设置WINE作为PE二进制的解释器：

0x01: 开启机制的方法

```
#mount binfmt_misc -t binfmt_misc
/proc/sys/fs/binfmt_misc
```

0x02: 添加新格式文件规则

```
#echo ":windows:M::MZ::/usr/bin/wine:" >
/proc/sys/fs/binfmt_misc/register
```

PE应用移植总结与几点建议

分类	产品	领域	优势	不足
同架构、不同操作系统平台间移植	wine	*NA*	**开源（LGPL v2）且可商用**	开源版本对性能和安全问题投入有限
	crossover (codeweavers)	面向企业客户提供ExecMode服务	基于wine的商业软件；提供了更加用户友好的方式在Linux和MacOS上运行Windows程序且宣传支持原生速度运行window软件	
	Proton (valve)	视频游戏类	基于wine的商业软件；旨在使Steam游戏平台上的Windows游戏能够在Linux系统上运行。提供了一种在Linux上运行Windows游戏的解决方案;	
不同架构+不同操作系统平台间移植；	box86/64	*NA*	**开源（MIT）且可商用**	原生效率的约55%
	wow64 (Arm版Win10)	消费级产品 (suface X)	MS官方发布，Arm sq1/2->win10 支持x86翻译为arm64指令，支持已翻译代码缓存和再运行时优化以提高性能；	
	Exagear (huawei-QCon2021)	企业级数据中心应用	支持x86(32/64)+arm32在arm64平台上指令动态翻译； 覆盖windows、android模拟； 宣称经优化后可以接近到aarch64 native的75%；	

基于欧拉社区支持PE应用现状的建议

I. 支持32位编译与运行时库

II. 扩展WINE外围功能及工具

III. PE应用迁移案例与常见问题整理归纳

THANKS

THANKS

THANKS

THANKS