

---

Subject: Re: Protect package - A starting copy protection system

Posted by [Tom1](#) on Fri, 08 Jun 2018 10:42:52 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Hi Max,

Thanks!

Here's an ELF -extended version of the instruction set detection:

```
unsigned short machine=0; // Unknown
if((* (unsigned int*)~buf)==0x464c457f) machine=*(unsigned short*)&buf[0x12]; // ELF machine ID
else if((* (unsigned short*)~buf)==0x5a4d){ // DOS header
    int coffindex=*(unsigned int*)&buf[0x3c]; // Coff header
    machine=*(unsigned short*)&buf[coffindex+4]; // Machine ID
}

switch(machine){
case 0x03: // x86 ELF
    Cout() << "Processing 32-bit i386 ELF\n";
    XED.Set32bitMode();
    break;
case 0x3e: // x86-64 ELF
    Cout() << "Processing 64-bit AMD64 ELF\n";
    XED.Set64bitMode();
    break;
case 0x14c: //i386 PE
    Cout() << "Processing 32-bit i386 COFF/PE\n";
    XED.Set32bitMode();
    break;
case 0x8664: // AMD64 PE
    Cout() << "Processing 64-bit AMD64 COFF/PE\n";
    XED.Set64bitMode();
    break;
default:
    Cout() << "Unknown executable - Cannot process\n";
    return;
}
```

My Linux VM running U++ is not quite healthy at the moment, so I could not test this ELF thing immediately, but maybe you can. Anyway, the ELF header info is from:

[https://en.wikipedia.org/wiki/Executable\\_and\\_Linkable\\_Format](https://en.wikipedia.org/wiki/Executable_and_Linkable_Format)

I can't show my actual code, but I'll see what I can do to demonstrate the optimization issue with a test case.

Thanks and best regards,

Tom

---