ATIS - Software Reverse Engineering

Prof. Dr. Christian Dietrich
dietrich@internet-sicherheit.de

2 Exercise: Assembly

2.1 Terminologie und Register

a. Nennen Sie die general purpose Register (GPR) der x86/IA-32-Architektur.

b. Was ändert sich hinsichtlich der GPR bei der 64-bit Erweiterung für x86, die auch häufig amd64 oder x86-64 bezeichnet wird?

c. Was ist die Funktion des Registers mit dem Namen `eip`? Worauf zeigt es?

d. Was ist das zero flag (ZF)? Zu welchem "Register" gehört es?

e. Wie ist der Bezeichner für das zweitniedrigste Byte des Registers `rax`?

2.2 Virtueller Speicher

a. Wie viel Hauptspeicher lässt sich mit einer 32-bit Architektur adressieren?

b. Nennen Sie Unterschiede zwischen Heap und Stack. Was ist das Besondere am Stack?

c. Wie lauten die Instruktionen, um Speicherworte (DWORD oder QWORD) auf den Stack zu legen und wieder zu entfernen?

d. Welchen Seiteneffekt hat die x86-64 Instruktion, die ein QWORD auf den Stack legt?

2.3 x86 Instruktionen

Die folgenden Aufgaben sollen das Verständnis von x86 Instruktionen für 32-bit und 64-bit vertiefen. Oft gibt es hierbei mehrere Lösungsmöglichkeiten. Nennen Sie mindestens eine. Falls Sie mehrere Lösungsarten kennen, vergleichen Sie sie.

a. Was ist ein Opcode? Was ist ein Mnemonic? Nennen Sie jeweils ein Beispiel.

b. Mit welchen Instruktionen lassen sich Daten von einer Hauptspeicheradresse zu einer anderen Hauptspeicheradresse kopieren?

c. x86-64: Mit welcher Instruktion lässt sich das Register `rax` auf Null setzen?

d. x86-64: Was ist der Wert von `rax` nachdem folgender Assembly-Code ausgeführt wurde?

```
1 mov rax, 0x1234
2 shr rax, 8
```

e. x86-64: Wie sieht es nach folgendem Code aus?

```
1 mov rax, 0x1234
2 shr rax, 0x40
```

f. x86-64: Erläutern Sie folgenden 32-bit x86 Code. Was geschieht hier? Wo könnte dieser Code Verwendung finden?

```
1 cmp word ptr [eax], 0x50545448
```

g. x86: Welchen Wert hat `eax` nach folgendem Code? Gehen Sie davon aus, dass `ebx` den Wert 0xC012 beinhalte.

```
1 lea eax, [ebx-0x100]
```

h. x86: Schreiben Sie den Code so um, dass Sie nur die Mnemonics `mov` und `sub` verwenden.

i. x86: Betrachten und annotieren Sie folgenden Code. Was passiert in der 4. Instruktion (Adresse 0x401027)? Was erwarten Sie?

```
1 Address         Hex dump        Command
2 00401018        BD 00204000     mov ebp, 0x402000
3 0040101D        BC 00114000     mov esp, 0x401100
4 00401022        68 00124000     push 0x401200
5 00401027        5C              pop esp
6 00401028        90              nop
```

2.4 x86 Assembly-Code

a. Schreiben Sie Assembly-Code, der prüft, ob der Inhalt des Registers `ecx` negativ ist. Falls ja, soll `eax` auf 1 gesetzt werden, sonst auf 0.

b. Es ist nicht erlaubt, den Instruction Pointer (`eip` bzw. `rip`) auszulesen. Überlegen Sie einen Workaround, der die Speicheradresse des ausführenden Codes liefert und in `eax` ablegt. Hinweis: Welche Instruktion hat als Seiteneffekt, dass die Adresse der nächsten Instruktion im Speicher abgelegt wird?