Name:_____

Written

[10] 1. Suppose a naïve virus scanner relies solely on string — or byte sequence — matching for its signatures. Give 2 techniques a virus author might use to avoid detection, and explain how they work.

[10] 2. From the course notes, list the properties of a good *watermark*.

Name:_____

3. Consider the following C code output by some decompiler for a program you are evaluating for security *on Linux amd64*:

```
1:
2:  int unk_func(int param1)
3:
4:  {
5:      int iVar1;
6:      ssize_t sVar2;
7:      int param1_local;
8:      char local3 [1022];
9:      ushort local2;
10:
11:     local2 = 0;
12:     sVar2 = recv(param1,&local2,2,0x100);
13:     if (sVar2 < 0) {
14:         iVar1 = 0;
15:     }
16:     else if (local2 < 0x401) {
17:         sVar2 = recv(param1,local3,(long)(int)(local2 - 2),0x100);
18:         if (sVar2 < 0) {
19:             iVar1 = 0;
20:         }
21:         else {
22:             iVar1 = FUN_00101070(local3);
23:         }
24:     }
25:     else {
26:         iVar1 = 0;
27:     }
28:     return iVar1;
29: }
30:
```

[2] (a) Suggest another name for `param1`.

[2] (b) Suggest another name for `local2`.

[2] (c) Suggest another name for `local3`.

[5] (d) Diagram the intended format of the message received by this function.

[5] (e) Locate and describe at least one vulnerability in the function, and briefly outline how it might be exploited. You may assume this is a system with no mitigations (i.e., DEP/NX bit, stack guards, ASLR, etc.)

[4] (f) Give two alternative programming recommendations to patch the vulnerability.

Name:_____

4. Consider the following disassembly:

```
0000000000101120:  f3 0f 1e fa    ENDBR64
0000000000101124:  eb 01          JMP     00101127
0000000000101126:  0f 48 8d 35    CMOVS   ECX,dword ptr [RBP + 0xed235]
                   d2 0e 00
000000000010112d:  00 bf 02 00    ADD     byte ptr [RDI + 0x2],BH
                   00 00
0000000000101133:  31 c0          XOR     EAX,EAX
0000000000101135:  e9 16 ff ff    JMP     00101050
                   ff
```

[2] (a) What approach does this tool use to produce its disassembly?

A. Linear Sweep B. Recursive Traversal C. Not enough information

[2] (b) Which instruction is the first incorrect instruction displayed in this listing?

[8] (c) Explain how you determined your answer to part (b).

[8] (d) Explain why the other disassembly approach would not fall for this obfuscation trick.

Name:_____

Practical

5. You should have a copy of the `practical_malware` executable *for Linux amd64*. It is a "malware" sample, but for simplicity's sake, it has only one rather innocuous command. However, it does employ some more sophisticated anti-RE techniques. **Choose 2** of the following 3 parts to complete:

   (a) It employs a time-based anti-debugging technique.

   [10] i. Explain how its implementation of the technique works.

   [10] ii. Patch the executable to defeat the technique. Submit your patched copy with your exam.

   (b) It connects to a remote command-and-control server.

   [4] i. What is the hostname and port of the server?

   [8] ii. Describe the protocol, i.e., what messages can it send and receive, and in what sequence?

   [8] iii. Explain why the first message it receives is so critical.

   (c) It decodes a portion of itself during execution.

   [5] i. What is the "key" used to protect and decode this portion?

   [10] ii. Decode and disassemble the protected portion. Submit a copy of the disassembly with your exam.

   [5] iii. What does the protected portion do?