Problem Sheet #2

Problem 2.1: stack smashing attack protection

Buffer overflows and more specifically stack smashing attacks are the most widely used exploits to attack systems. A number of techniques have been proposed to prevent or at least detect such attacks.

The paper by Cowan et. al. [1] proposes a compiler technique which aims to detect stack smashing attacks while the paper by Baratloo etc. al. [2] describes two libraries that provide protection against stack smashing attacks.

Network Intrusion Detection Systems (IDSs) try to detect attack codes in the network by identifying attack signatures. More recent attack strings use polymorphic attack codes and are therefore harder to detect by IDSs. The paper by Pasupulati et. al. [3] describes an extension for the open network intrusion detection system called snort which can detect polymorphic attack codes.

Your task is to read the papers and to answer the following questions.

a) Briefly summarize the techniques described in [1] and [2] and evaluate their strong and weak points.

b) Paper [2] outlines a specific situation where they use trap instructions. Why are they needed and how do they solve the problem?

c) Papers [1] and [2] both mention the proposal to make stacks non-executable in the related work section. Both papers argue that this won’t work since an executable stack is commonly used for a) function trampolines, b) signal handling and c) functional programming languages. Write a short explanation how and why an executable stack is used for these three purposes.

d) Paper [3] shows the following snort configuration line.

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS any 
    (msg:"MSSQL2000 remote UDP exploit"; 
     range:"42ae1000-42b0caa4"; 
     dsize:475<>550; 
     rangeoffset:97; 
     rangedepth:20;)
```

Explain which checks will be performed on a TCP test that matches the selection criteria.

References

