**REVERSE ENGINERRING BOOK FOR HACKERS**

**INTRODUCTION:-**

**In the concept of security holes in program and protocols,**

**We have two problems here:-**

**1) The increasing size and complexity of the internet.**

**2) The sensitivity of the information stored through the internet has created a rich environment for our next generation hackers**

**If you are infected by malware 🡪 why do not reverse engineering it**

**Also for hackers 🡪 if there is a research in a program to find a vulnerable function in it …..**

**Why do not reverse engineering it to find this vulnerable function.**

**An overview of code debuggers**

**Sooner or later you will want to know everything about an exe file. Even if you perform a vulnerability research on Microsoft**

**Sequel server you will deal with exe file.**

**In hacking we want to know :-**

1. **The exact memory address that is calling**
2. **The exact region of memory location that is writing to**
3. **What region it is reading from**
4. **What registers it is making of**

**In reverse engineering an exe file we do not have the source code but our helper here is the Debugger.**

**Debuggers will help you in reverse engineering of files even**

**If you do not have the source code**

**The debugger will help in :**

1. **Reverse engineering malwares**
2. **Reverse engineering programs to find a vulnerability in**

**It in exploits researches**

**We will use in this book 2 different debuggers**

**.Community debuggers**

**& .IDA pro debuggers**

**With IDA pro you can reverse engineering any type of application file..**

**1. Console application such as Xbox , Playstation**

**2. MAC computer systems**

**3. Windows 4. Unix 5. And many other**

**In debuggers you can see:-**

**.Memory calls that is occurred**

**.Windows DLL that is being called**

**Assembly language for hackers**

**We will use here the visual studio c# and system programming with assembly language for our goal**

**To reverse engineering malwares and to RE program files**

**To find vulnerable functions and even have some short**

**Idea of the steps to develop exploits for this vulnerability**

**Assembly and the IA-32 processors**

**If we take instance the code:-**

**Return 0;**