

# Malicious Code Detection

**Rootkit :: a ghost in the shell**

Helen Welch

Dr Joohan Lee

Research Experience for Undergraduates

Summer 2004

# Outline of Presentation

---

- Terminology defined
- Rootkits defined
- Introduction to Rootkits
- Overview of the NT Rootkit
- Vanquish Rootkit
- Conclusion

# Terminology [8]

---

**Malicious Code** is a catch all term that refers to any type of computer software that cause harm to a computer system

**Virus** is a program (a block of executable code) which attaches itself to, overwrites or otherwise replaces another program in order to reproduce itself without the knowledge of the PC user

**Trojan Horse** is a program intended to perform some covert and usually malicious act which the victim did not expect or want

**Worm** is a program which spreads (usually) over network connections. Unlike a virus, it does not attach itself to a host program.

# What is a Rootkit?

---

“ a set of programs which \*PATCH\* and \*TROJAN\* existing execution paths within a system”

- Greg Hoglund

# What is a Rootkit?

---

- Programs that help attackers keep their position as root
- “root” is the highest level of administration on a UNIX based platform
- “kit” is a collection of tools
- “Rootkits contain tools which help attackers hide their presence as well as give the attacker full control of the server or host continuously without being noticed” [5]

# Intro to Rootkits

---

- Modify core system utilities in order to hide the presence of the intruder and their tools
- May disable auditing when certain users are logged on
- Allow anyone to log in if a backdoor is used
- Patch the kernel itself, allowing anyone to run privileged code

# Intro to Rootkits

---

Two types of rootkits:

## 1. Application rootkits [5]

- Established at the application layer
- Replaces well known system binary files (ls, netstat, killall) with “Trojanned” ones
- The trojanned system files will help hide the attackers presence, report false information and provide a backdoor for the attacker

# Intro to Rootkits

---

Two types of rootkits:

## 2. Kernel rootkits [5]

- Established at the kernel level
- Extremely hard to detect and remove
- Works by exploiting and manipulating kernel capabilities

# Intro to Rootkits

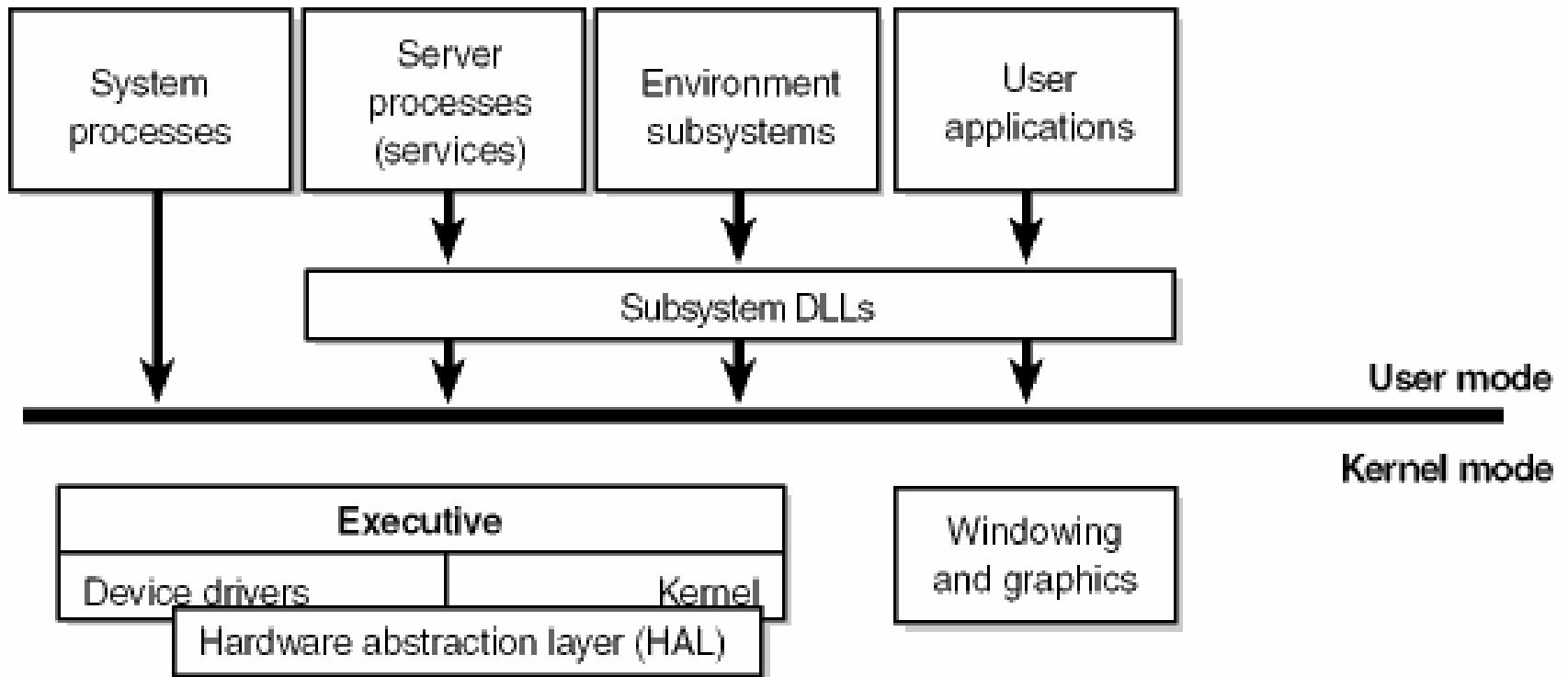
---

## Elements that make up a rootkit

- Backdoors
- Sniffers
- Cleaners (Log Bashers)
- etc

# NT Rootkit

authored by Greg Hoglund



## Diagram of Windows NT/2000 Architecture [1]

(Reproduced from Microsoft Press Online)

- The traditional ways kernel-mode rootkits hide various objects include [7]
  - hooking dynamic link libraries (DLL's) functions (API/Import Address Table hooking),
  - modifying DLL's functions (raw code change),
  - hooking Service Table,
  - hooking interrupt descriptor table,
  - and modifying kernel code
- The end result of these techniques is the ability to be able to control process calls and protect system resources [7]

# Vanquish Rootkit

v0.2.0

authored by XShadow

# Vanquish Rootkit [6]

---

- Vanquish is a DLL injection based Romanian rootkit that hides files, folders, registry entries and logs passwords
- It works by hooking API calls and processes them.
- Designed to work under Windows XP
- Vanquish DLL is internally divided into six submodules
  1. DllUtils
  2. HideFiles
  3. HideReg
  4. HideServices
  5. PwdLog
  6. SourceProtect

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 1. DllUtils:

- » Inject Vanquish DLL into new processes
- » Make sure nobody will unload Vanquish DLL

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 2. HideFiles:

- » Hide files/folders containing the magic string “vanquish“
- » A hidden file/folder won't get reported as occupying space and cannot be found with "Search For Files or Folders..." or similar, and a folder containing hidden files/folders cannot be erased.

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 3. HideReg :

- » Hide registry entries containing the same magic string
- » Uses an advanced cached system to keep track of enumerated keys/values and hide the ones that need to be hidden, but keeping the high-performance of registry

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 4. HideServices :

- » Hide service entries containing the magic string in their name
- » Uses the same logic as in HideReg but without the complicated caching system

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 5. PwdLog :

- » Logs username, passwords and domain
- » First it intercepts calls to LogonUserW (probably 'runas' commands, switch user and similar). To get the logon prompt password we hook the msgina dll function WlxLoggedOutSAS. The passwords are present in the logfile.

# Vanquish Rootkit [6]

---

- Vanquish DLL is internally divided into six submodules

## 6. SourceProtect :

- » Prevent deletion of certain files/folders
- » Prevent change of system time

# Conclusion

---

- “It is an ‘arms race’ between attackers and computer security specialists and we suggest using Intel’s x86 architecture to its fullest potential by using all four hardware privilege modes and by placing critical operating system components within the most secure level will prevent modification of operating system components, ensuring it continues to function as specified.” [2]
- Continuing research into detecting hidden processes and channels that may be used by the attacker [2]

# Works Cited

---

- [1]“Beware of Geeks Bearing Gifts: A Windows NT Rootkit Explored”.  
Cibelli, Fredric J. 4 April 2001.
- [2]“HIDDEN PROCESSES: The Implication of Instruction Detection”.  
Butler, Undercoffer, Pinkston. June 2003.
- [3]Phrack Magazine. “A Real NT Rootkit, patching the NT Kernel”.  
Hoglund, Greg. 9 Sept 1999.
- [4]Phrack Magazine. “Execution path analysis: finding kernel based  
rootkits”. Rutkowski, Jan.
- [5] [www.rootkits.com](http://www.rootkits.com). July 2003.
- [6] Vanquish Readme.txt
- [7]“Windows Rootkits”. Gaydosh, Adam. 24 March 2004.
- [8][http://www.itsc.state.md.us/oldsite/info/InternetSecurity/BestPractices/  
Viruses.htm](http://www.itsc.state.md.us/oldsite/info/InternetSecurity/BestPractices/Viruses.htm)