SANS GCIH CERTIFICATION GUIDE:
Created by Michael LaSalvia 2/2010

Hosted on: http://www.digitaloffensive.com

BOOK 504.1

A. **Incident Handling Process 6 steps** (Preparation, Identification, Containment, Eradication, Recovery, Lessons Leaned)
B. **What is incident handling?** An action plan for dealing with the misuse of a computer systems and network.
C. **What is an event?** Any observable occurrence in a system and / or network.
D. **What is an incident?** Is an adverse event in an information system / and or network.

1. Preparation Page 15
2. Identification Page 46
   - Cheat sheets: Page 58
3. Containment Page 92
4. Eradication Page 111
5. Recovery Page 117
6. Lessons Learned Page 121
7. Incident Tips Page 130
   - a. Espionage Page 131
   - b. Unauthorized use Page 138
   - c. Insider Threat Page 151
   - d. Intellectual Property Page 164
     - i. Patent Page 167
     - ii. Copyright Page 168
     - iii. Fair use Page 169
     - iv. Trademark/Servicemark Page170
     - v. Trade Secrets Page 173
     - vi. PICERL for intellectual Property Page 174-179
8. Law, Crime and Evidence Page 180
   - a. Criminal vs. Civil Page 182
   - b. Arrest/False Arrest Page 183
   - c. Search/Seizure with and without a warrant Page 184-185
   - d. US Cyber Crime Laws Page 187-189
     - i. Cyber Security Enhancement Act of 2002
       1. Title 18 sec 1362: Prohibits malicious injury or destruction of com equip
       2. Title 18 sec 2510: wire & electronic inter and inter of oral coms,
       3. Title 18 sec 1030 Computer fraud financial (MONEY) government , foreign

SANS GCIH CERTIFICATION GUIDE:

BOOK 504.2

Trends:

1. Hacktivism:	Page 11	**3.** Software Distro Site Attacks	Page 13-14
2. Attack for fun and profit:	Page 12	**4.** The Golden Age	Page 15

Reconnaissance:

1. **Domain Name Registration** (Address, Phone, Contacts, Authoritative DNS) Page 19
   a. Useful for SE, War Dialing and scanning
2. **Whois**: Allows you get information on domains and IP. Including nameservers Page 20-21
   a. **Defense** is to just deal with it.
   b. **Identification** impossible
3. **DNS interrogation**: Uses information from a whois to pull additional info. Page 26-30
   a. **Defense**: Use split DNS (internal and external), limit zone transfers, harden servers
   b. **Identification**: look for zone transfers
4. **Web Site Searches**: Search targets site, search job sites, search partner sites, search social media sites, blogs and newspapers. Press releases, contacts, design docs and so on Page 32-35
   a. **Defense:** limit what is posted, generalize job openings, and protect directories from crawlers.
   b. **Identification:** Search for crawler traffic and mass site downloads
5. **Google**: Johnny Long and GHDB. Use to find vulnerabilities Page 37-48
      i. **Defense**: robots.txt (NOINDEX, NOFOLLOW, NOSNIPPET, NOARCHIVE) removal of content and re-crawl site google.com/addurl.html . Conduct self searches.
   b. Phonebook searches (phonebook: and REVERSE:). **Removal** /help/pbremoval.html
   c. Google Maps (View physical security of a building, roads, doors & so on.
   d. Search directives:
      i. site, link, intitle, inurl, info, cache, filetype and ext (the same, better to just use doc, pdf & so on), (-) and word (+) and word (.) wild card for a single character
   e. Automated Google **w/ Key**: Site digger and Wikto / **Without** Goolag, Wikto w/ AURA & SecApp GHDB
6. **Maltego**: intelligence gathering tool by, maps relationships using transforms Page 50-52
   a. **Defense**: make sure your data is accurate and scan yourself. Ask that inaccurate / damaging data be removed.
7. **War Dialers**: dials number looking for modems and secondary dial tone. Page 56-64
   a. THC Scan (newest version can be sued on botnet)
   b. Warvox: Uses voip accounts can do 1,000 numbers an hour, spoof caller ID and call as self.
   c. Use the results to try to access systems

8. **War Driving / wireless:** Page 66-81
    a. **Netstumbler**: limited driver support, relies on SSID, Active, GPS tie in.
    b. **Wellenreiter**: Passive scanning, packet capture, IP gathering, Linux
    c. **Cracking & Sniffing:** Kismet, ominpeek, aircrak-ng, wepCrack, ASLEAP, CowPatty
    d. **Karma:** pretends to be everything, responds to all probe requests, allows you to act as requested resource can be tied into metasploit.
    e. **Defense:** WPA or better, mac address filtering, Non attractive SSID or no SSID, use a vpn tunnel, better placement of AP, look for rouge devices, wireless IPS / IDS (ARUBA, Motorola)
9. **Network Mapping / Nmap:** Tracert, traceroute and nmap, zenmap gui Page 85-94
    a. **IP Headet:** TTL, SRC IP and DST IP
    b. **Traceroute:** Uses low TTL and ICMP time exceed message to map. Increases each by 1 after a time exceed till it hits host.
    c. **Nmap:** Now uses PN (NO PING), Sends 4 packets to check if host is up ICMP ECHO Request, ICMP Timestamp request, TCP SYN to port 443 and TCP ACK to 80 if running as UID 0 or if not then syn.
        i. More efficient mapping of larger networks using. Starts with large TTL and will adjust till it find the correct TTL and then starts counting backwards.
    d. **Zenmap:** Visual Graphing of the network map based on the results from nmap.
    e. **Defense:** Disable incoming ICMP echo requests and outbound time exceeded.
10. **Port Scanning/ Active OS**: Nmap, Xprob2 Page 95-120
    a. Nmap scan type Page 101
    b. Namp IP Spoofing and Idle Scan: IP Identification field, predictable Page 105-108
    c. Active OS Finger printing Page 111-113
    d. Tools: netstat, fport, wmic, sc, netstat and checkconfig Page 115-119
    e. **Defense:** turn off service not needed, stateful firewall and proxy, IPS/ IDS, Change OS identification info.
11. **Passive OS**: P0F2: Uses a sniffer and database for matching, defense above Page 126-128
12. **Firewalk** allows you to determine what ports are open on a firewall Page 130-136
13. **Fragmentation Attacks**: breaking up a packet to bypass IDS Page 137-145
    a. Tiny fragmentation
    b. Overlapping fragmentation
14. **Fragrouter & Fragroute:** tools too fragment packets and bypass IDS/IPS Page 146-148
    a. **Defense:** reassemble packets before IPS ?IDS, host based IPS/ IDS, Keep up to date, make sure your IPS/IDS properly speced.
15. **Vulnerability Scanning:** Nessus, SATAN and so on, mostly NESSUS info Page 151-164
16. **Web: CGI, PHP, JSP, ASP:** Nikto scanner, Whisker, IDS Invasion Page 165-178
    a. GET Request: passing parameters values on the url
    b. POST Request: passing parameters in the body
    c. **Defense:** Run server with least privilege, Remove default scripts and directories, Patch and harden, Good code (scrub bad parameters)
17. **Null Sessions:** Enum, net use, net view, winfingerprint, smbclient Page 179-210

SANS GCIH CERTIFICATION GUIDE:

BOOK 504.3

1. **IP Address Spoofing:** Page 5-15
   a. Change the IP: incomplete handshake, good for a DOS
   b. TCP sequence # guessing: Requires you to knock the spoofed IP off line and guess sequence #. Good for trust relations on Linux such as R services.
   c. Source Routing spoofing: A router on the path to victim must allow source routing. NC can do source routing.
   d. **Defense:** Anti spoofing enable, disable source routing
2. **Netcat (nc):** Swiss army knife, multiple version & variations. Like Linux cat Page 16-48
   a. Client mode: nc IP 22
   b. Listen mode: nc –l –p 22
   c. Netcat command switches: Page 20
   d. Transferring files with Netcat Page 21-22
   e. Vulnerability scanning and port scanning Page 23
   f. Backdoors, persistent backdoor & reverse shells (-e) Page 25-27
   g. Relays: windows use a bat file and linux use backpipe Page 28-30
   h. Exercise and examples Page 35-48
   i. **Defense:** Know what is on your system, filter ports, close un needed services.
3. **Sniffers:** Passive = Wireshark, Active = Dsniff Page 49-75
   a. **Hub** = broadcast, traffic to all ports || **Switch** = uses cam and arp to match physical port and IP.
   b. **Arp** Maps IP (network layer) to Mac (data link layer)
   c. **Dsniff Components**: dsniff, arpsoof, macof, tcpkill and so on Page 54
   d. **Gratuitous ARPS:** send a arp response without a request, arp cache poisoning.
   e. **Macof:** flood switch bogus MAC addresses, trying to fill CAM table to cause the switch to become like a hub. Or to confuse the switch that two ports are the same machine.
   f. **Arpspoof:** Uses are arp cache poisoning by sending false ARP messages into a LAN.
   g. **Dsniff:** tcpkill, tcpnice, filesnarf, mail, url, and msgsnarf, webspy Page 60-62
   h. **MITM:** DNSpoof, WEbmitm, SSHmitm, SSLstrip Page 63-70
   i. **Defense:** hard code ARP table on important LAN's, lock ports to mac, use ssh v2, use encryption on network,
      a. **Detect:** local: ifconfig on kernel 2.4 and earlier, ip link: kernel 2.4 or later, promqry and a few others. Remotely: EtherARP, Sentinel. Warning messages from SSL and SSH, messed up arps
4. **Session Hijacking:** Uses spoofing and Sniffing. Session based protocol Page 77-86
   a. Finding a session and using tcp sequence to hijack as session
   b. Ack storms get created while they try to figure what is going on. Take out src or use arp cache poisoning.
   c. Ettercap Page 81

5. **Arp and mac Exercise** Page 90-96
6. **DNS cache poisoning:** 3 ways. Kaminsky the best Page 97-111
7. **Buffer Overflow:** due to not properly checking data input Page 113-129
    - a. Step 1: Find potential overflows: search code for weak func Page 121
        - **a.** Google Code Search & micorosft !exploitable tool
        - **b.** Cram input: Search input: A= 0x41: ABDEF Page 125
    - b. Step 2: Push exploit code into mem:
        - **a.** Small machine code, tailored to processor, watch for null char
    - c. Step 3: Setting the return pointer: Hardest part
        - **a.** Analyze the code
        - **b.** Guess
        - **c.** Use NOP sleds: better chances that your code will be executed.
8. **Metasploit:** Frame work for exploiting and development Page 131-150
    - a. Meterpreter: hides in exploited process, multi-purpose Page 136-137
    - b. Routines for development: find the exact RP, msfelfscan and msfpescan to check exe and libraries for signs of vulnerabilities.
    - c. **Defense of buffer overflows:** non executable stack and DEP Page142-150
        - i. Safe and secure code development.
9. **File & protocol parser overflows:** Page 151-155
10. **Format String Attacks:** caused by no format string in printf, snprintf, sprint Page 157-183
    - a. Curious user input %x %d %n,
        - **a.** %x print hexadecimal value
        - **b.** %n prints the value of user input
        - **c.** %d decimal interger
    - b. Format string attacks push to the stack in reverse order
        - **a.** **Little endian** = Intel: \xc0\xfa\xff\xbf = 0xbffffac0
    - c. Allows you to write anywhere in memory, overwrite user credentials and so on.
    - d. **Defense:** Apply patches, safe programing practices, src code review.
    - e. **Format string exercise:** Page 183-209

SANS GCIH CERTIFICATION GUIDE:

BOOK 504.4:

1. **Password Cracking:** protect from unauthorized disclosure, modification, removal Page 5-52
   a. **Password Representations** are stored hashed or encrypted passwords. **Windows** = SAM **Linux** = /etc/shadow
2. **Password Guessing:** use a valid ID and try a list of passwords, no brute force, slow Page 6
3. **THC Hydra:** Password guessing, dictionary support, many protocols Page 7
4. **Password Cracking:** Determine the password w/ just the cipher text password rep Page 8-13
   a. **Dictionary Attack:** Fastest method uses a list of words (dictionary), also checks concatenation of words.
   b. **Brute Force:** Trys every possible combination, guarantee to crack dependent on time and encryption algorithm.
   c. **Hybrid:** builds on dictionary by adding #'s and symbols to dictionary words like password1
   d. Password cracking is good for auditing and recovering, get permission. Don't use for migrating users.
5. **LANMAN Hashes:** Found on win NT/2K/XP/2003 Page 15-17
   a. **Very weak**
      i. 14 char or less passes are hashed. NO Salts
      ii. Padded to exactly 14 char and all upper case
      iii. Split the 14 char into two 7 char strings, each 7byte string is a DES key
      iv. The empty pad is AAD3B43 (shows in cain for passwords that are less then 8)
      v. Hybrid attacks in Cain or other tools work the best.
      vi. ALT char makes it take longer months or years
6. **NT Hashes:** Better then LANMAN, Upper &Lower Case, hashed using MD4 Page 18
7. **LANMAN and NT Hashes:**
   a. Users with identical passwords have same hash, use precomputed dictionary
8. **Salts:** Random number used to seed crypto algorithm Page 19
   a. **Windows:** Don't use salts so hashes are the same
   b. **Linux:** Uses salts: salt =random,password/salt hash =value
      i. Salt =vqQO0mlr, password/salt hash =JvrqDBUVi7jYU6Ddr7G2, STORE: 1 vqQO0mlr JvrqDBUVi7jYU6Ddr7G2
      ii. $= delimited, $1 = md5 , $8 byte salt, $encrypted salted password
9. **Pre-generated Tables:** Rainbow tables, MD5 crack, already has hashes Page 21
10. **Cain & Abel:** Is two tools, they are feature rich, Cain collects & Abel is a remote Page 22-28
    a. **Abel:** Is a remote tool almost like a backdoor (dump remote password hashes)
    b. **Cain:** collects a lot of information, includes the ability to crack passwords, arp cache poison, sniffer and much more.
       i. **Features:** Page 24
       ii. **Cracks:** LANMAN, NT HASH, **MORE ON** Page 26
       iii. Cain supports Rainbow tables for cracking using winrtgen.exe, dictionary, simple hybrid and brute force attacks

11. **Obtaining hashes:** Page 29-30
12. **Defense:** Page 31-34
    - a. **Disable LANMAN:** Regkeys Page 32
    - b. **Password Enforcement:** Group policy Page 33
    - c. **SYSkey:** Adds an additional 128-bin strong encryption to the SAM Database Page 34
13. **John the Ripper:** Very fast password cracker focus on Linux but can do windows Page 36-42
    - a. Supports many algorithms
    - b. You must feed it a encrypted password file
    - c. To use the shadow file you must unshado it and combine the /etc/passwd and shadow
        - i. Unshadow /etc/passwd /etc/shadow > combined
            - 1. Feed john the combined file.
    - **d.** **Cracking modes:** Single Crack, Wordlist, Incremental, External Page 40
    - **e.** John auto supports and detects: BSDI extended DES, FreeBSD MD5, OpenBSD blowfish, lanman
        - **i.** Additional patches are available for other algorithms
    - **f.** Cracked passwords are stored in file john.pot
14. **Unix Password file and Shadow file:** Page 38
15. **Pass the Hash Attack:** use the stolen hash instead of cracking it for the password Page 53-56
    - **a.** Good for lsass, smb, LANMAN challenge response, NTLM1 and 2
    - **b.** PSHtoolkit: For windows Page 55
    - **c.** For linux modified samba code from JoMo-Kun and Foofus Page 55
16. **Worms:** Spread over the network & Self replicate Page 58-80
    - **a.** Take over one system and turn that system into an attacker as well.
    - **b.** Worms been around for decades: Morris worm 1988
    - **c.** **Multi Exploit worms:** Page 61
    - **d.** **Multi Platform worms** Page 62
    - **e.** **Zero day worms** Page 63
    - **f.** **Warhol / Flash** Prescan large amounts of exploitable hosts ie.10,000 first 10,000 infections take seconds. Each infection scans for new vulnerable machines.
    - **g.** **Polymorphic:** Page 66-67
        - **i.** **Admutate:** by k2
    - **h.** **Metamorphic worms:** change appearance and functions Page 69
    - **i.** **Ethical Worms:** using fast moving worms to patch systems. Cause legal issues.
17. **The rise of the bots:** spread through worms, email, bundled software, droppers, + Page 72-80
    - **a.** **Communication:** over IRC, Social sites, websites, p2p, waste, non standard irc port
    - **b.** **Fast Flux:** Uses round robin DNS to point to victims that have web proxies that redirect to the real evil host.
    - **c.** **Phatbot:** Page 77-79
18. **Defense:** Patch, encrypt hard drive
19. **Virtual Machines:** Vmcat, Truman, red pill, Scoopy Page 82-88
    - **a.** Important to make you code run differently to avoid analyst, or guess system escaping to host.
    - **b.** **Local VME detection:** Page 83
    - **c.** **Remote VME detection:** Page 84
    - **d.** **VME escape** Page 85
20. **Cracking Web Apps:** OWASP

21. **Account Harvesting:** Using error messages or URL's to determine valid user ID's. Page 92-95
    - **a.** Error might say invalid user or invalid pass. Key is to say either or and not to give the attacker the ability to differentiate.
22. **SQL Injection:** Structured Query Language attacks Page 98-107
    - a. Must identify a user input field that is vulnerable. Start by adding string quotation characters to the input fields. Look for errors that can help you execute SQL injection such as database names, table names and so on.
    - b. **Characters:** (--) (;)(*)(%)(_) or 1=1,SELECT, JOIN, UPDATE Page 100
    - c. **Finding SQL errors:** Page 101
    - d. **Dropping Data** Page 102
    - e. **Grabbing more data** Page 103
    - f. **Getting database structures** Page 104
    - g. **Defense:** Sanitize user inout, limit application access to database, mod_security, stored procedures, WAF
23. **Cross Site Scripting**: XSS: based on a web app that reflects user input back to a user Page 109-123
    - a. Usually JavaScript or VBS is inserted into a user field and the outcome is reflected back to the user. Or it can be placed in a url as a variable.
    - b. **Launching attack**: email, forums, websites, spread the url
    - c. **Cookie stealing**: Site must be vulnerable to xss, due to domain objects.
    - **d. Harvest browser history**
    - **e. Conduct network scans / reconfigure routers**
    - **f. Exploit Administrative apps**
    - **g. Defense:** sanitize user input, turn off browser scripting, mod_security, noscript, WAF
24. **Attacking State:** Tracking sessions and altering variables or state to change data Page 125-141
    - **a. URL Session tracking:** Session ID is in the URL Page 126-127
    - **b. Hidden Form elements:** in the code of the page. Save a local copy and edit it Page 126-127
    - **c. Cookies:** Open up and edit. Page 126-127
    - **d.** SSL and non persistent cookies do not protect session tracking Page 128-129
    - **e.** Browser addins and Proxies to alter HTTP requests Page 130-136
        - **i.** Tamper Data: Firefox addin
        - **ii.** Add N Edit cookies: Firefox addin
        - **iii.** Paros Proxy: feature rich proxy, SSL, SPIDER, DETECT UNSAFE, DEHASH Page 134
    - **f. Defense:** WAF, Use time stamps in session id, prevent collision in session id, digitally sign or use a keyed hash function, encrypt cookies
25. **Denial of Service:** local and remote, using up all available resources Page 143-186
    - **a. CPU HOG**: Sets itself at priority 16, forces tskmgr to increase others to 15 Page 146-147
    - **b. Rose**: Sends highly fragmented packets writing the last frag over and over, not packet flood attack ip stack Page 150-151
    - **c. SMURF Attacks**: uses broadcast address and spoofing to amplify attack Page 154-160
        - **i. Smurf and papa smurf**
        - **ii. Fraggle UDP version**
    - **d. DNS Amplification & EDNS:** uses large records to amplify dos send spoof small query and get large response back to the host. Page 163-168
    - **e. SYN Attack**: Attacker either does not respond to the syn-ack or spoofs the src, causing half open connections using up all the connections. Page 168-174
    - **f. DOS Tools** Page 176

g. **DDOS:** Use to use special tools, most are by botnets now Page 177-187

   i. **Reflected DDOS:** Using zombies and spoofing, legit site attacks victim

   ii. **Pulsing Zombies:** bots attack for short time then go idle

   iii. **HTTP Flooding:** Get request blend in

h. **Defense:** Patching, turn off un needed services, anti spoofing, disable ICMP at GW, IDS, block offending IP, egress filtering.

SANS GCIH CERTIFICATION GUIDE:

BOOK 504.5

1. **Backdoors & Trojans:** Page 6-9
   - **a.** **Trojan**: program that looks functional but is really sinister
   - **b.** **Backdoor:** a program that allows an attacker to bypass normal security controls on a system.
   - **c.** **Trojan Horse backdoor:** malicious programs can contain both
   - **d.** **Rootkit:** Alters the OS so it look normal but it is not.
2. **Malware Layers:** Page 7
   - **a.** **App Level Trojan horse backdoor:** Evil app installed (ivy, vnc, bots)
   - **b.** **User mode:** Critical OS components replaced (AFX rootkit, lrk6, Hacker Defender)
   - **c.** **Kernel Mode:** Kernel altered (KIS, FU, FUTo, super user control kit)
   - **d.** **Boot Sector:** malicious boot sector alters kernel as it is loaded (Vbootkit2.0, kon-boot)
   - **e.** **Firmware:** Malicious code loaded in firmware
   - **f.** **Malware Microcode:** Malicious CPU Microcode
3. **VNC:** Virtual network computing, made for legit use, though abused often Page14-18
   - a. Gui across the network over port 5900, client listens on 5500 when shoveling
   - b. Can also shovel a connection to a listening client
   - c. Multiple platform support and is used in metasploit
   - d. Server can run as a service or in app mode. Configure not to show in systray.
4. **Poison IVY:** Page 19
5. **Common remote control backdoor capabilities** Page 20-24
6. **Setri:** uses OLE to communicate with a hidden browser, if it has inet it will work Page 25-26
   - a. Do to using hidden browser it gets through firewalls, NAT's and proxies
   - b. Go through anonymizer and connection broker where scripts run
   - c. Many new malware is using this method.
7. **Defense:** Harden system, use updated AV tools, safeweb surfing, look for modified reg keys odd ports
8. **Wrappers & Packers:** used to hide malicious files Page 31-35
   - **a.** **Wrappers:** Also known as binders. Create backdoors by wrapping malicious app into a good program
     - i. Saranwrap, Elitewrap, Silkrope 2000 , AFX File Lace (encrypts as well) Trojan man (encrypts)
     - ii. Users install backdoor first and sees the actual program secondary
   - b. **Packers**: try to thwart reverse engineering or execution of the attack code without the attack doing it.
     - i. **Linux: burneye** (three layers of protection, obf, password, fingerprinting (tying to OS)
       1. **Burndump:** beats burneye for all modes except password.
     - ii. **Windows**: UPX , EXE32pack, ASPack, EXEstealth Page 34-35
       1. **Ollydbg**: with plugins can unpack many packers**.**

9. **Memory Analysis:** Must get a memory dump first: MemoryDD.bat, fastdump, win32dd Page 37-62
   a. **Volatile Framework**: Open source module written in python
      i. **Important modules** Page 38
      ii. **View connections:** python volatility connections –f path_to_dump Page 39
         1. **On live windows: netstat –nao | find “ESTABLISHED”**
      iii. **View Process:** python volatility pslist –f path_to_dump
         1. **On live windows:** wmic process get name,parentprocessid,processed
      iv. **View DLLs & Command Line:** python volatility dlllist –p [pid] –f path_to_dump
         1. **On live windows:** tasklist /m /fi “pid eq [pid]” and wmic process where processed=[pid] get commandline

USER MODE ROOTKITS: 66-82: (application Layer): Ring 3

10. **LRK Rootkit:** backdoors sshd & login programs Page 67-70
    a. Password set by attacker. When used accounting entries are not written.
    b. Password cant be found by strings
    c. Attack won’t show up in who command
    d. Backdoor components : **login, rshd, sshd, inetd, tcpd, chfn, chsh, password, su**
    e. Hiding: **ps, top, pidof, killall crontab, netstat, ifconfig, ls, find, du, syslogd**
11. **Linux Rootkit hiding evidence tools:**
    a. fix: modifies creation date
    b. wted: allows for editing wtmp & utmp
    c. z2: erases utmp, wtmp & lastlog
12. **Windows User mode rootkits:** DLL injection and API hooking. Attacker injects code in running process. Such as explorer.exe, windows gui Page 73-74
13. **AFX Windows root Kit:** injects itself it to running DLL or programs Page 75-79
    a. Attacker uses the config console to create executable, executable copied to target and ran.
    b. Newer version hiding is automatically configured
    c. Iexplore.dll and explorer.dll created, file copies over to system 32
    d. Hides processes and ports
14. **Preperation:** harden and patch system, Don’t let attacker get root in first place.
15. **Identification:** Difficult, can use tools like Tripwire and AIDE, use hashes to compare checksums on non writable medium. Echo * vs ls
16. **Containment:** Analyze other systems changes made by discovered root kits.
17. **Eradication:** Format the drive, reinstall and patch, change passwords
18. **Recovery:** Monitor system closely.

KERNEL MODE ROOTKITS 80-122 (run at kernel level and have much more power over the system)

19. **Kernel mode rootkits:**
    a. Don’t require modification to individual programs.
    b. Kernel mode is ring 0, relies on hardware level protection
    c. Fantasy worl hidden from administrator
20. **5 Types of Kernel Mode Root kits:**
    a. **Loadable Kernel modules:** (Unix) & Device Drivers (windows) ← Most Popular

b. **Altering Kernel in Memory:** /dev/kmem (holds map of kernel memory) Windows (system memory map): SUCKit for linux and FU for windows does this. Vista kernel by hogging mem and writing kernel pages to hard disk.

c. **Changing Kernel File on the hard drive:** /boot/vmlinuz on Unix and NTOSKRNl.exe and NTLDR on windows. On windows both must be altered as the NTLDR does checksum on the NTOSKRNL

d. **Virtualizing the system:** Joanna's **Blue Bill** uses the AMD virtualization instructions. **VT-x (Vitriol)** for intel. Attackers can put the machine in a virtual environment. **Runs entire kernel in user mode**

e. **Running programs directly in Kernel mode: KML** (Kernel Mode Linux), Windows NT rootkit does this. Very dangerous and can leave the system unstable. **Runs user mode in kernel mode**

21. **Adore:** Another Linux Kernel mode rootkit. Focus on hiding stuff kernel 2.4 & 2.6 Page 96-98

a. **Two Components:** Adore the LKM and AVA, the program that interacts with the LKM

b. **Adore Capabilities** Page 97

22. **KIS:** (Kernel Intrusion System): targets 2.2 & 2.4 kernel that use loadable kern mods Page 100-105

a. Receives command on network but don't listen on a port.

i. Comms on udp arbitrary ports grabbed by the kernel. Uses a sniffer

b. Configured and controlled with a GUI.

c. Features Page 101

d. Survives reboot by altering an executable such as init

e. Creates a hidden process and everything done via it is in the hidden process

23. **SInAr**: (Solaris 10 Kernel mode rootkit) Page 107-108

24. **FU:** Windows kernel mode root kit, name taken Linux SU command Page 110

a. **2000/XP/2003:** Available at www.rootkit.com

25. **FUTo:** Update to FU, extends original code. Page 111

a. Tries to dodge rootkit detection tools: Blacklight and Icesword

i. Blacklight and Icesword call openprocess api for all possible processids, if pid successfully open but the associated process cant be seen it alert possible rootkit.

b. FUTo removes reference to hidden process.

26. **Defenses:** Harden machines, Good Security Templates, AV, **Detection:** Chkrootkit (linux), Rootkit hunter (linux), Rootkit Revealer (Windows), Backlight, Icesword, Tripwire, Bootable Resposne CD's such as Helix, IDS / IPS

Covering Tracks in LINUX: 124-146

27. **Hiding Files:** simply name something with .name or .. name or even just “. ” (dot space),” .. “(dot, dot, space) or just “ “.

a. **ls –**a: will show the hidden file.

b. **They are usually stored in:** /dev , /tmp , /etc , /usr/local/man , /usr/src

28. **Editing Log files:** logs are in ASCII format and able to be edited by hand Page 128-129

a. Check /etc/syslog.conf for log paths

b. Common logs and logs of interest:

i. /var/log/secure

ii. /var/log/message

iii. /var/log/httpd/error_log and access_log

29. **Editing Shell history:** .bash_history: Contains the last N commands ran. Page 129-130

a. Some attackers add commands, most delete commands

   b. Writes commands to log after graceful shell log out
   c. So to avoid this ungracefully log out by killing the shell killall bash

30. **Linux accounting files:** Page 133-135
    a. **Utmp:** "who command" contains info about current users that are logged in. Default location /var/run/utmp
    b. **Wtmp:** contains data about past logins. Default location /var/log/wtmp
    c. **Btmp:** contains data about bad login attempts. Bad to use as it may contain passwords, if users are not careful. Default location /var/log/btmp. Almost never used
    d. **Lastlog:** shows login name, port and last login for each user. Default location /var/log/lastlog
    e. Cant be edited by hand (utmp, wtmp & btmp) Special tool like remove.c

Covering tracks in Windows

31. **Hiding Files in windows:** (NTFS) Page 148-150
    a. **Alternate data streams:** multiple streams can can be attached, hide malicious files in standard files. Hides size as well. Windows vista + gives ability to see them using **dir /r.** Linux can see them as well using smb and ADSs
    b. **To hide:** type hackstuff.exe > notepad.exe:stream1 or cp hackstuff.exe notepad.exe:stream1.exe
    c. **To extract:** cp notepad.exe:stream1.exe hackstuff.exe
    d. **Attach to directory:** notepad <file_or_directory_name>:<sctream_name>
    e. **LADS:** Allows you to see them in windows
    f. **Streams and Streams Shell extension**
32. **Log editing in windows:** Default location %root%\SYSTEM32\CONFIG Page 153-158
    a. **Event log files are:**
       i. AppEvent.EVT
       ii. SecEvent.Evt
       iii. SysEvent.Evt
    b. Attackers with admin access can delete logs fully or over fill logs with bogus info.
    c. With physical access attackers can use a linux boot cd to edit the log file
    d. **WinZapper:** edits windows logs on NT 4 and 2k, works on xp and 2003 but a bit buggy.
    e. **Meterpreter: clearev command:** clears all logs
33. **Defense:**
    a. **Preparation:** log to remote server, burn logs on a schedule, snare or kiwi to syslog for windows, encrypt logs
    b. **Identification:** look for gaps or corrupt logs

Covering tracks on the Network: Tunneling and covert channels Page 169-204

34. **Reverse WWW Shell:** Client / Server, Client installed on victim Page 170-171
    a. Src port is 1024 dst port is 80, looks like outbound web surfing, bypasses firewall. Uses http get
    b. Can use credentials
    c. Connects to Attackers server and they will have a command line
    d. Requires perl, could be rewritten.
    e. Similar tool is sneakin, that looks like telnet.
35. **ICMP Tunnels:** Page 174-176
    a. **LOKI** – Linux Shell
    b. **ICMPShell** – Linux

c. **PingChat** – Windows Chat

d. **ICMPCmd** – Windows cmd

e. **Ptunne**l: Windows and Linux, TCP over ICMP echo and reply

  i. Has a client and proxy

  ii. Configure client with a port to get data from and a ultimate dest address

    1. Attacker makes connection to a the local port → data Is sent to the proxy over ICMP and then to the final dst over TCP

36. **Covert Channels:** Page 178-195