Strike/Forensics Checklist (Initial Vulnerability)

|  |  |
| --- | --- |
| No. | Control |
| 1 | Initial System Vulnerability Scan:   * Linux/UNIX   + <http://www.openvas.org/> ⇒ OpenVAS is a framework of several services and tools offering a comprehensive and powerful vulnerability scanning and vulnerability management solution. The framework is part of Greenbone Networks' commercial vulnerability management solution from which developments are contributed to the Open Source community since 2009.     - How to get OpenVas web interface running:<https://www.digitalocean.com/community/tutorials/how-to-use-openvas-to-audit-the-security-of-remote-systems-on-ubuntu-12-04>   Greenbone Security Assist: <http://docs.greenbone.net/src/gsa/7.0/index.html>   * + - This is very powerful tool but it is really hard to setup and configure. If links provided are not enough just google it. You will get a lot of assistance out there.   + [Lynis](https://cisofy.com/lynis/) - Lynis is an open source security auditing tool. Used by system administrators, security professionals, and auditors, to evaluate the security defenses of their Linux and Unix-based systems. It runs on the host itself, so it performs more extensive security scans than vulnerability scanners.   Quick how to run Lynis scan:   * **git clone** [**https://github.com/CISOfy/Lynis**](https://github.com/CISOfy/Lynis) * **cd lynis** * **./lynis audit system -Q**   Lynis Get Started Guide: <https://cisofy.com/documentation/lynis/get-started/>   * Installation:   + **yum install lynis** or   + **git clone** [**https://github.com/CISOfy/lynis**](https://github.com/CISOfy/lynis)+ **cd lynis** * Running Lynis:   + **./lynis**   + **lynis audit system** - a basic scan without any preconfiguration   + **lynis audit system --quick** - running without pauses * Example Commands:   Command Description  **audit system** Perform a system audit  **show commands** Show available Lynis commands  **show help**  Provide a help screen  **show profiles** Display discovered profiles  **show settings** List all active settings from profiles **show version** Display current Lynis version   * Options:   Option Abbreviated Description  **--auditor "Given name Surname"** Assign an auditor name to the audit (report)  **--cronjob** Run Lynis as cronjob (includes -c -Q)  **--debug** Show debug information, useful for troubleshooting and development  **--help**  **-h**  Shows valid parameters  **--man-page** View man page  **--no-colors** Do not use any colors  **--pentest** Perform a penetration test scan (non-privileged)  **--quick**  **-Q**  Don't wait for user input, except on errors  **--quiet**  **-q**  Only show warnings (includes --quick, but doesn't wait)  **--reverse-colors**  Use a different color scheme for lighter backgrounds  **--verbose**  Show more screen output   * Tips:   + If Lynis is not installed as package (with included man page), use **--man** or **nroff -man ./lynis.8**   + For systems where the shell background is light, use **--nocolors** or **--reverse-colors**   + Use command **show options** to see all available parameters of Lynis * Windows (not for Windows 10)   + <https://msdn.microsoft.com/en-us/library/ff647642.aspx> ⇒ Microsoft Baseline Security Analyzer (MBSA) checks for available updates to the operating system, Microsoft Data Access Components (MDAC), MSXML (Microsoft XML Parser), .NET Framework, and SQL Server. MBSA also scans a computer for insecure configuration settings. When MBSA checks for Windows service packs and patches, it includes in its scan Windows components, such as Internet Information Services (IIS) and COM+. MBSA uses Microsoft Update and Windows Server Update Services (WSUS) technologies to determine needed updates. This Microsoft Update data source is obtained either directly from the Microsoft Update Web site or, if offline or in a secure environment, from an offline catalog file named Wsusscn2.cab.     - This one is not the best tool to guarantee security but most software for windows vulnerability scans are commercial so this one is probably the best we can do. Better way to enhance windows security would be manually going through all the configuration details that systems team provides for Windows OS and having secure configurations for the running services.     - Download Link : <https://www.microsoft.com/en-us/download/details.aspx?id=7558>     - Double click to open MBSA. Click "Scan a computer".     - Make sure the options “**Check for Windows Administrative vulnerabilities**”, “**Check for weak passwords**” and “**Check for security updates**“ are checked. You can uncheck the options “**Cech for IIS vulnerabilities**” and “**Check for SQL vulnerabilities**” if you don’t have them installed.   How to interpret the MBSA scan report:   * Administrative vulnerability check:   + Red X - critical check failed (black password)   + Yellow X - non-critical check failed (password that doesn’t expire)   + Green check - check passes   + Blue asterisks - for more info about the computer * Security update checks:   + Red X - security update missing   + Yellow X - warning (not the latest update)   + Green check - good   + Blue asterisks - update not available (not approved on the Update Services server) * You can click “**Result details**” for info on how to fix it. |
| 2 | Volatile Memory Analysis:   * Volatility ⇒ The Volatility Foundation is an independent 501(c) (3) non-profit organization. The foundation’s mission is to promote the use of Volatility and memory analysis within the forensics community, to defend the project’s intellectual property (trademarks, licenses, etc.) and longevity, and to help advance innovative memory analysis research.   + Download Link : <http://www.volatilityfoundation.org/releases>   Ubuntu: **apt-get install volatility**  Git repo: **git clone https://github.com/volatilityfoundation/volatility.git**  How to use Volatility: <https://www.howtoforge.com/tutorial/how-to-install-and-use-volatility-memory-forensic-tool/>  Volatility Analysis:   * + Identify the supported “profiles”:   **volatility-2.4.standalone.exe imageinfo -f memdump3.raw**   * + Get a list of running process in the memory dump examples with profile = Win7SP0x86:   **volatility-2.4.standalone.exe --profile=Win7SP0x86 pslist -f memdump3.raw**   * + Display the processes in tree (parent/child) format. Identify the parent process of the malicious program:   **volatility-2.4.standalone.exe --profile=Win7SP0x86 pstree -f memdump3.raw**   * + The following command with malfind switch is used to dump the malicious DLL's in “output” directory. (Run with PID of all suspicious processes; Process ID: 1120 (svchost.exe))   **volatility-2.4.standalone.exe --profile=Win7SP0x86 malfind -D E:\output/pid-1120 -p 1120 -f memdump3.raw**  No malicious DLL found:    Found malicious DLL:    Example output dump:    The dumps of the malicious programs are scanning using Windows defender and Malware bytes.  Command history (CMD history):   * + Scan for the history of commands run on the machine:   **volatility-2.4.standalone.exe --profile=Win7SP0x86 cmdscan -f memdump3.raw**  Network Connection:   * + As we know that network connection can be found in the memory analysis, so “netscan” plug-in is run against the memory image and the result shows that malicious program “iexplore” opened several connections on the victim machine.   **volatility-2.4.standalone.exe" --profile=Win7SP0x86 netscan -f memdump3.raw**    General Info and commands:  Used: git clone <https://github.com/volatilityfoundation/volatility.git>  Basic Usage: **python vol.py [plugin] -f [image] --profile=[profile]**  Example: python vol.py pslist -f /path/to/memory.img --profile=Win7SP1x64   * + List of Plugins/Readme: <https://github.com/volatilityfoundation/volatility/blob/master/README.txt>   + GitHub repo with installation instructions : <https://github.com/volatilityfoundation/volatility/wiki/Installation>   + More installation instructions : <https://www.howtoforge.com/tutorial/how-to-install-and-use-volatility-memory-forensic-tool/>   + Interesting Usages: <http://resources.infosecinstitute.com/memory-forensics-and-analysis-using-volatility/#gref> |
| 3 | Network:  Universal Tools: [Nmap](https://nmap.org/); [Nessus](https://www.tenable.com/products/nessus-vulnerability-scanner) (free of charge for personal use in a non-enterprise environment)  Nmap:   * Check if Nmap is already installed:   **nmap --version**   * Download: <https://nmap.org/download.html>   Nmap cheat sheet:  **Basic Scanning Techniques**  Scan a single target —> **nmap [target]**  Scan multiple targets —> **nmap [target1,target2,etc]**  Scan a list of targets —-> **nmap -iL [list.txt]**  Scan a range of hosts —-> **nmap [range of IP addresses]**  Scan an entire subnet —-> **nmap [IP address/cdir]**  Scan random hosts —-> **nmap -iR [number]**  Excluding targets from a scan —> **nmap [targets] –exclude [targets]**  Excluding targets using a list —> **nmap [targets] –excludefile [list.txt]**  Perform an aggressive scan —> **nmap -A [target]**  Scan an IPv6 target —> **nmap -6 [target]**  **Discovery Options**  Perform a ping scan only —> **nmap -sP [target]**  Don’t ping —> **nmap -PN [target]**  TCP SYN Ping —> **nmap -PS [target]**  TCP ACK ping —-> **nmap -PA [target]**  UDP ping —-> **nmap -PU [target]**  SCTP Init Ping —> **nmap -PY [target]**  ICMP echo ping —-> **nmap -PE [target]**  ICMP Timestamp ping —> **nmap -PP [target]**  ICMP address mask ping —> **nmap -PM [target]**  IP protocol ping —-> **nmap -PO [target]**  ARP ping —> **nmap -PR [target]**  Traceroute —> **nmap –traceroute [target]**  Force reverse DNS resolution —> **nmap -R [target]**  Disable reverse DNS resolution —> **nmap -n [target]**  Alternative DNS lookup —> **nmap –system-dns [target]**  Manually specify DNS servers —> **nmap –dns-servers [servers] [target]**  Create a host list —-> **nmap -sL [targets]**  **Advanced Scanning Options**  TCP SYN Scan —> **nmap -sS [target]**  TCP connect scan —-> **nmap -sT [target]**  UDP scan —-> **nmap -sU [target]**  TCP Null scan —-> **nmap -sN [target]**  TCP Fin scan —> **nmap -sF [target]**  Xmas scan —-> **nmap -sX [target]**  TCP ACK scan —> **nmap -sA [target]**  Custom TCP scan —-> **nmap –scanflags [flags] [target]**  IP protocol scan —-> **nmap -sO [target]**  Send Raw Ethernet packets —-> **nmap –send-eth [target]**  Send IP packets —-> **nmap –send-ip [target]**  **Port Scanning Options**  Perform a fast scan —> **nmap -F [target]**  Scan specific ports —-> **nmap -p [ports] [target]**  Scan ports by name —-> **nmap -p [port name] [target]**  Scan ports by protocol —-> **nmap -sU -sT -p U:[ports],T:[ports] [target]**  Scan all ports —-> **nmap -p “\*” [target]**  Scan top ports —–> **nmap –top-ports [number] [target]**  Perform a sequential port scan —-> **nmap -r [target]**  **Version Detection**  Operating system detection —-> **nmap -O [target]**  Submit TCP/IP Fingerprints —->**<http://www.nmap.org/submit/>**  Attempt to guess an unknown —-> **nmap -O –osscan-guess [target]**  Service version detection —-> **nmap -sV [target]**  Troubleshooting version scans —-> **nmap -sV –version-trace [target]**  Perform a RPC scan —-> **nmap -sR [target]**  **Timing Options**  Timing Templates —-> **nmap -T [0-5] [target]**  Set the packet TTL —-> **nmap –ttl [time] [target]**  Minimum of parallel connections —-> **nmap –min-parallelism [number] [target]**  Maximum of parallel connection —-> **nmap –max-parallelism [number] [target]**  Minimum host group size —–> **nmap –min-hostgroup [number] [targets]**  Maximum host group size —-> **nmap –max-hostgroup [number] [targets]**  Maximum RTT timeout —–> **nmap –initial-rtt-timeout [time] [target]**  Initial RTT timeout —-> **nmap –max-rtt-timeout [TTL] [target]**  Maximum retries —-> **nmap –max-retries [number] [target]**  Host timeout —-> **nmap –host-timeout [time] [target]**  Minimum Scan delay —-> **nmap –scan-delay [time] [target]**  Maximum scan delay —-> **nmap –max-scan-delay [time] [target]**  Minimum packet rate —-> **nmap –min-rate [number] [target]**  Maximum packet rate —-> **nmap –max-rate [number] [target]**  Defeat reset rate limits —-> **nmap –defeat-rst-ratelimit [target]**  **Firewall Evasion Techniques**  Fragment packets —-> **nmap -f [target]**  Specify a specific MTU —-> **nmap –mtu [MTU] [target]**  Use a decoy —-> **nmap -D RND: [number] [target]**  Idle zombie scan —> **nmap -sI [zombie] [target]**  Manually specify a source port —-> **nmap –source-port [port] [target]**  Append random data —-> **nmap –data-length [size] [target]**  Randomize target scan order —-> **nmap –randomize-hosts [target]**  Spoof MAC Address —-> **nmap –spoof-mac [MAC|0|vendor] [target]**  Send bad checksums —-> **nmap –badsum [target]**  **Output Options**  Save output to a text file —-> **nmap -oN [scan.txt] [target]**  Save output to a xml file —> **nmap -oX [scan.xml] [target]**  Grepable output —-> **nmap -oG [scan.txt] [target]**  Output all supported file types —-> **nmap -oA [path/filename] [target]**  Periodically display statistics —-> **nmap –stats-every [time] [target]**  133t output —-> **nmap -oS [scan.txt] [target]**  **Troubleshooting and debugging**  Help —> **nmap -h**  Display Nmap version —-> **nmap -V**  Verbose output —-> **nmap -v [target]**  Debugging —-> **nmap -d [target]**  Display port state reason —-> **nmap –reason [target]**  Only display open ports —-> **nmap –open [target]**  Trace packets —> **nmap –packet-trace [target]**  Display host networking —> **nmap –iflist**  Specify a network interface —> **nmap -e [interface] [target]**  **Nmap Scripting Engine**  Execute individual scripts —> **nmap –script [script.nse] [target]**  Execute multiple scripts —-> **nmap –script [expression] [target]**  Script categories —-> **all, auth, default, discovery, external, intrusive, malware, safe, vuln**  Execute scripts by category —-> **nmap –script [category] [target]**  Execute multiple scripts categories —-> **nmap –script [category1,category2, etc]**  Troubleshoot scripts —-> **nmap –script [script] –script-trace [target]**  Update the script database —-> **nmap –script-updatedb**  **Ndiff**  Comparison using Ndiff —-> **ndiff [scan1.xml] [scan2.xml]**  Ndiff verbose mode —-> **ndiff -v [scan1.xml] [scan2.xml]**  XML output mode —-> **ndiff –xml [scan1.xm] [scan2.xml]**  Other:   * Wireshark   Useful Wireshark filters:  Display Filters  **ip.addr == 10.0.0.1** - Sets a filter for any packet with 10.0.0.1, as either the source or dest  **ip.addr==10.0.0.1 && ip.addr==10.0.0.2** - sets a conversation filter between the two defined IP addresses  **http or dns** - sets a filter to display all http and dns  **tcp or udp** -sets a filter to display all http and dns  **tcp.port==4000** - sets a filter for any TCP packet with 4000 as a source or dest port  **tcp.analysis.flags** - filter for all tcp packets with flags (no responce etc.)  **tcp.flags.reset==1** - displays all TCP resets  **http.request** - displays all HTTP GET requests  **http.response** - displays all HTTP GET response  **tcp contains traffic** - displays all TCP packets that contain the word ‘traffic’. Excellent when searching on a specific string or user ID  **!(arp or icmp or dns)** - masks out arp, icmp, dns, or whatever other protocols may be background noise. Allowing you to focus on the traffic of interest  **udp contains 33:27:58** - sets a filter for the HEX values of 0x33 0x27 0x58 at any offset  **tcp.analysis.retransmission** - displays all retransmissions in the trace. Helps when tracking down slow application performance and packet loss  **tcp.flags.syn==1** - show only tcp SYN packets  **tcp.flags.reset==1** - show all tcp resets  **sip && rtp** - show all Session Initiation Protocol packets (Internet telephone calls, multimedia distribution, and multimedia conferences) and Real-time Transport (transmitting audio, video or simulation data)   * Network Miner * Xplico |
| 4 | Malware:   * [Chkrootkit](http://www.chkrootkit.org/): a shell script that checks system binaries for  rootkit modification (for Linux)   \* ifpromisc.c: checks if the network interface is in promiscuous  mode.   \* chklastlog.c: checks for lastlog deletions.   \* chkwtmp.c: checks for wtmp deletions.   \* check\_wtmpx.c: checks for wtmpx deletions. (Solaris only)   \* chkproc.c: checks for signs of LKM trojans.   \* chkdirs.c: checks for signs of LKM trojans.   \* strings.c: quick and dirty strings replacement.   \* chkutmp.c: checks for utmp deletions.   * + Download: <http://www.chkrootkit.org/download/>   + Installation:   To compile the C programs type: **make sense** After that it is ready to use and you can simply type: **./chkrootkit**   * + Usage:   chkrootkit must run as root. The simplest way is:  **./chkrootkit**  This will perform all tests. You can also specify only the tests you  want, as shown below:   Usage: **./chkrootkit [options] [testname ...]**  Options:  -h show this help and exit  -V show version information and exit  -l show available tests  -d debug  -q quiet mode  -x expert mode  -r dir use dir as the root directory  -p dir1:dir2:dirN path for the external commands used by chkrootkit  -n skip NFS mounted dirs Where testname stands for one or more from the following list:  aliens asp bindshell lkm rexedcs sniffer w55808 wted scalper slapper z2 chkutmp amd basename biff chfn chsh cron crontab date du dirname echo egrep env find fingerd gpm grep hdparm su ifconfig inetd inetdconf identd init killall ldsopreload login ls lsof mail mingetty netstat named passwd pidof pop2 pop3 ps pstree rpcinfo rlogind rshd slogin sendmail sshd syslogd tar tcpd tcpdump top telnetd timed traceroute vdir w write  Example Uses:   * + Checks for trojaned ps and ls binaries and also checks if the network interface is in promiscuous mode.   **./chkrootkit ps ls sniffer**   * + The `-q' option can be used to put chkrootkit in quiet mode -- in  this mode only output messages with `infected' status are shown.   + With the `-x' option the user can examine suspicious strings in the  binary programs that may indicate a trojan -- all the analysis is  left to the user.   + Lots of data can be seen with:   **./chkrootkit -x | more**   * + Pathnames inside system commands:   **./chkrootkit -x | egrep '^/'**   * + To use, for example, binaries in /cdrom/bin:   **./chkrootkit -p /cdrom/bin**   * + It is possible to add more paths with a `:'   **./chkrootkit -p /cdrom/bin:/floppy/mybin**   * + Specify a new rootdir with the `-r' option.For example, suppose the disk you want to check is mounted under  /mnt, then:   **./chkrootkit -r /mnt**  Output Messages:  The following messages are printed by chkrootkit (except with the -x  and -q command options) during its tests:   * + **"INFECTED"**: the test has identified a command probably modified by a known rootkit;   + **"not infected"**: the test didn't find any known rootkit signature.   + **"not tested"**: the test was not performed -- this could happen in  the following situations:  a) the test is OS specific;  b) the test depends on an external program that is not available;  c) some specific command line options are given. (e.g. -r ).   + **"not found"**: the command to be tested is not available;   + **"Vulnerable but disabled"**: the command is infected but not in use.  (not running or commented in inetd.conf) |
| 5 | File Carving/Parsing/Metadata:   * AnalyzeMTF - A Python Script analyzeMFT.py is designed to fully parse the MFT file from an NTFS file system. (git repo: <https://github.com/dkovar/analyzeMFT> )   Installation =========== You should now be able to install analyzeMFT with pip:   pip install analyzeMFT   Alternatively:   git clone<https://github.com/dkovar/analyzeMFT.git>  cd analyzeMFT  python setup.py install (or, just run it from that directory)  Usage =========== Usage: analyzeMFT.py [options]  Options:  -h, --help show this help message and exit  -v, --version report version and exit   File input options:   -f FILE, --file=FILE read MFT from FILE  File output options:   -o FILE, --output=FILE write results to FILE  -c FILE, --csvtimefile=FILE write CSV format timeline file  -b FILE, --bodyfile=FILE write MAC information to bodyfile  Options specific to body files:   --bodystd Use STD\_INFO timestamps for body file rather than FN  timestamps  --bodyfull Use full path name + filename rather than just  filename  Other options:   -a, --anomaly turn on anomaly detection  -l, --localtz report times using local timezone  -e, --excel print date/time in Excel friendly format  -d, --debug turn on debugging output  -s, --saveinmemory Save a copy of the decoded MFT in memory. Do not use for very large MFTs  -p, --progress Show systematic progress reports.  -w, --windows-path Use windows path separator when constructing the filepath instead of linux  Output =========  analyzeMFT can produce output in CSV or bodyfile format.  CSV output --------- The output is currently written in CSV format. Due to the fact that Excel automatically determines the type of data in a column, it is recommended that you write the output to a file without the .csv extension, open it in Excel, and set all the columns to "Text" rather than "General" when the import wizard starts. Failure to do so will result in Excel formatting the columns in a way that misrepresents the data.  I could pad the data in such a way that forces Excel to set the column type correctly but this might break other tools.  GUI: You can turn off all the GUI dependencies by setting the noGUI flag to 'True'. This is for installations that don't want to install the tk/tcl libraries.   * [Exiftool](http://www.sno.phy.queensu.ca/~phil/exiftool/) - reading, writing and editing meta information in image, audio and video files. * [Foremost](http://foremost.sourceforge.net/) - Foremost is a console program to recover files based on their headers, footers, and internal data structures. * [RegRipper](https://github.com/keydet89/RegRipper2.8) - For Windows Registry Hives |
| 6 (Monitoring) | Auditing: System helps system administrators create an audit trail, a log for every action on the server. We can track security-relevant events, record the events in a log file, and detect misuse or unauthorized activities by inspecting the audit log files.   * Windows : <https://technet.microsoft.com/en-us/library/dd277403.aspx> * Windows 2000 server:   Viewing the Security Log:   1. Open Shared Folders. Click **Start**, select **Programs**, select **Administrative Tools**, click **Computer Management**. 2. In the console tree, click **Event Viewer**. Double-click **Security** and in the details pane, examine the list of audit events  * Linux/UNIX: <https://www.isaca.org/Journal/archives/2015/Volume-4/Pages/auditing-linux-unix-server-operating-systems.aspx> |