|  |
| --- |
| SERVER HARDENING IN WEB APPLICATION |
|  |

Server Hardening is the process of enhancing server security through a variety of means which results in a much more secure server operating environment. This is due to the advanced security measures that are put in place during the server hardening process.

**Server Hardening**, probably one of the most important tasks to be handled on your servers, becomes more understandable when you realize all the risks involved. The default configuration of most operating systems are not designed with security as the primary focus. Instead, default setups focus more on usability, communications and functionality. To protect your servers, you must establish solid and sophisticated server hardening policies for all servers in your organization. Developing a server hardening checklist would likely be a great first step in increasing your server and network security. Make sure that your checklist includes minimum security practices that you expect of your staff. If you go with a consultant, you can provide them with your server hardening checklist to use as a baseline.

**Server Hardening** **Tips & Tricks:**

Every server security conscious organization will have their own methods for maintaining adequate system and network security. Often you will find that server hardening consultants can bring your security efforts up a notch with their specialized expertise.  
  
**Some common server hardening tips & tricks include:**  
  
-> Use Data Encryption for your Communications  
- > Avoid using insecure protocols that send your information or passwords in plain text.  
- > Minimize unnecessary software on your servers.  
- > Disable Unwanted SUID and SGID Binaries  
- > Keep your operating system up to date, especially security patches.  
- > Using security extensions is a plus.  
- > When using Linux, SELinux should be considered. Linux server hardening is a primary focus for the web hosting industry, however in web hosting SELinux is probably not a good option as it often causes issues when the server is used for web hosting purposes.

- > User Accounts should have very strong passwords  
- > Change passwords on a regular basis and do not reuse them  
- > Lock accounts after too many login failures. Often these login failures are illegitimate attempts to gain access to your system.  
-> Do not permit empty passwords.  
- > SSH Hardening  
- > Change the port from default to a non-standard one  
- > Disable direct root logins. Switch to root from a lower level account only when necessary.  
- > Unnecessary services should be disabled. Disable all instances of IRC - BitchX, bnc, eggdrop, generic-sniffers, guard services, ircd, psyBNC, ptlink.  
- >Securing /tmp /var/tmp /dev/shm  
- >Hide BIND DNS Sever Version and Apache version  
- >Hardening sysctl.conf  
- > Server hardening by installing Root Kit Hunter and ChrootKit hunter.  
-> Minimize open network ports to be only what is needed for your specific circumstances.  
- > Configure the system firewall (Iptables) or get a software installed like CSF or APF. Proper setup of a firewall itself can prevent many attacks.  
- > Consider also using a hardware firewall  
-> Separate partitions in ways that make your system more secure.  
- > Disable unwanted binaries  
- > Maintain server logs; mirror logs to a separate log server  
-> Install Log watch and review log watch emails daily. Investigate any suspicious activity on your server.  
-> Use brute force and intrusion detection systems  
- > Install Linux Socket Monitor - Detects/alerts when new sockets are created on your system, often revealing hacker activity  
-> Install Mod\_security as Webserver Hardening  
- > Hardening the Php installation  
- >Limit user accounts to accessing only what they need. Increased access should only be on an as-needed basis.  
- > Maintain proper backups  
- > Don't forget about physical server security.

What is BIOS

BIOS (basic input/output system) is the program a personal computer's [microprocessor](https://whatis.techtarget.com/definition/microprocessor-logic-chip) uses to get the computer system started after you turn it on. It also manages data flow between the computer's [operating system](https://whatis.techtarget.com/definition/operating-system-OS) and attached devices such as the [hard disk](https://searchstorage.techtarget.com/definition/hard-disk), [video adapter](https://whatis.techtarget.com/definition/video-adapter), [keyboard](https://whatis.techtarget.com/definition/keyboard), [mouse](https://whatis.techtarget.com/definition/mouse) and [printer](https://whatis.techtarget.com/definition/printer).

BIOS is an integral part of your computer and comes with it when you bring it home. (In contrast, the operating system can either be pre-installed by the manufacturer or vendor or installed by the user.) BIOS is a program that is made accessible to the microprocessor on an erasable programmable read-only memory ([EPROM](https://whatis.techtarget.com/definition/EPROM)) chip. When you turn on your computer, the microprocessor passes control to the BIOS program, which is always located at the same place on EPROM.

When BIOS boots up (starts up) your computer, it first determines whether all of the attachments are in place and operational and then it loads the operating system (or key parts of it) into your computer's random access memory ([RAM](https://searchstorage.techtarget.com/definition/RAM-random-access-memory)) from your hard disk or diskette drive.With BIOS, your operating system and its applications are freed from having to understand exact details (such as hardware addresses) about the attached input/output devices. When device details change, only the BIOS program needs to be changed. Sometimes this change can be made during your system setup. In any case, neither your operating system or any applications you use need to be changed.

Although BIOS is theoretically always the intermediary between the microprocessor and I/O device control information and data flow, in some cases, BIOS can arrange for data to flow directly to memory from devices (such as video cards) that require faster data flow to be effective.

**Booting Process**

Booting (also known as booting up) is the initial set of operations that a computer system performs when electrical power is switched on. The process begins when a computer that has been turned off is re-energized, and ends when the computer is ready to perform its normal operations. On modern general purpose computers, this can take tens of seconds and typically involves performing power-on self-test, locating and initializing peripheral devices, and then finding, loading and starting an operating system. Many computer systems also allow these operations to be initiated by a software command without cycling power, in what is known as a soft reboot, though some of the initial operations might be skipped on a soft reboot. A boot loader is a computer program that loads the main operating system or runtime environment for the computer after completion of self-tests.

The computer term boot is short for bootstrap or bootstrap load and derives from the phrase to pull oneself up by one’s bootstraps. The usage calls attention to the paradox that a computer cannot run without first loading software but some software must run before any software can be loaded. Early computers used a variety of ad-hoc methods to get a fragment of software into memory to solve this problem. The invention of integrated circuit Read-only memory (ROM) of various types solved the paradox by allowing computers to be shipped with a start up program that could not be erased, but growth in the size of ROM has allowed ever more elaborate start up procedures to be implemented.

There are numerous examples of single and multi-stage boot sequences that begin with the execution of boot program(s) stored in boot ROMs. During the booting process, the binary code of an operating system or runtime environment may be loaded from nonvolatile secondary storage (such as a hard disk drive) into volatile, or random-access memory (RAM) and then executed. Some simpler embedded systems do not require a noticeable boot sequence to begin functioning and may simply run operational programs stored in read-only memory (ROM) when turned on.

**The order of booting –**

In order for a computer to successfully boot, its BIOS, operating system and hardware components must all be working properly; failure of any one of these three elements will likely result in a failed boot sequence.When the computer’s power is first turned on, the CPU initializes itself, which is triggered by a series of clock ticks generated by the system clock. Part of the CPU’s initialization is to look to the system’s ROM BIOS for its first instruction in the startup program. The ROM BIOS stores the first instruction, which is the instruction to run the power-on self test (POST), in a predetermined memory address. POST begins by checking the BIOS chip and then tests CMOS RAM. If the POST does not detect a battery failure, it then continues to initialize the CPU, checking the inventoried hardware devices (such as the video card), secondary storage devices, such as hard drives and floppy drives, ports and other hardware devices, such as the keyboard and mouse, to ensure they are functioning properly.

Once the POST has determined that all components are functioning properly and the CPU has successfully initialized, the BIOS looks for an OS to load.

The BIOS typically looks to the CMOS chip to tell it where to find the OS, and in most PCs, the OS loads from the C drive on the hard drive even though the BIOS has the capability to load the OS from a floppy disk, CD or ZIP drive. The order of drives that the CMOS looks to in order to locate the OS is called the boot sequence, which can be changed by altering the CMOS setup. Looking to the appropriate boot drive, the BIOS will first encounter the boot record, which tells it where to find the beginning of the OS and the subsequent program file that will initialize the OS.

Once the OS initializes, the BIOS copies its files into memory and the OS basically takes over control of the boot process. Now in control, the OS performs another inventory of the system’s memory and memory availability (which the BIOS already checked) and loads the device drivers that it needs to control the peripheral devices, such as a printer, scanner, optical drive, mouse and keyboard. This is the final stage in the boot process, after which the user can access the system’s applications to perform tasks.

**UEFI**

Short for Unified Extensible Firmware Interface, UEFI is a specification that defines a more modernized model for the interface between computer [operating systems](https://www.webopedia.com/TERM/O/operating_system.html) and platform [firmware](https://www.webopedia.com/TERM/F/firmware.html) during the boot, or start-up, process.

UEFI originated as the Intel Boot Initiative in the late 1990s before being turned over to the Unified EFI Forum, and today the forum and specification remain the result of a collaborative effort between computer processor manufacturers like AMD and Intel and software operating system companies like Microsoft and Apple.    
 In many ways, UEFI serves as a software-driven, bare-bones operating system that can sit on top of the legacy [BIOS](https://www.webopedia.com/TERM/B/BIOS.html) boot process, and like BIOS, UEFI is responsible for initializing the hardware of a device or computer before passing control of the hardware to the operating system. Most newer computer platforms support both UEFI and legacy [BIOS](https://www.webopedia.com/TERM/B/BIOS.html) booting in order to ease the transition to UEFI and accommodate older operating systems that don't have built-in UEFI support.

The UEFI specification offers advanced features over BIOS such as secure boot, low-level cryptography, network authentication and universal graphics drivers. The Secure Boot functionality in UEFI provides the basis for the [Microsoft Secure Boot](https://www.webopedia.com/TERM/M/microsoft_secure_boot.html) feature in [Windows 8](https://www.webopedia.com/TERM/W/windows_8.html) that enables the OS to detect [rootkits](https://www.webopedia.com/TERM/R/rootkit.html) and similar [malware](https://www.webopedia.com/TERM/M/malware.html) attacks.

Differents Between LVM and RAID

|  |  |  |
| --- | --- | --- |
| **S.No.** | **RAID** | **LVM** |
| 1. | RAID is used for redundancy. | LVM is a way in which you partition the hard disk logically and it contains its own advantages. |
| 2. | A RAID device is a physical grouping of disk devices in order to create a logical presentation of one device to an Operating System for redundancy or performance or a combination of the two. | LVM is a logical layer that that can be anipulated in order to create and, or expand a logical presentation of a disk device to an Operating System. |
| 3. | RAID is a way to create a redundant or striped block device with redundancy using other physical block devices. | LVM usually sits on top of RAID blocks or even standard block devices to accomplish the same result as a partitioning, however it is much more flexible than partitions. You can create multiple volumes crossing multiple physical devices, remove physical devices without loosing data, resize the volumes, create snapshots, etc |
| 4. | RAID is either a software or a hardware technique to create data storage redundancy across multiple block devices based on required RAID levels. | LVM is a software tool to manage large pool of storage devices making them appear as a single manageable pool of storage resource. LVM can be used to manage a large pool of what we call Just-a-bunch-of-Disk (JBOD) presenting them as a single logical volume and thereby create various partitions for software RAID. |
| 5. | RAID is NOT any kind of Data backup solution. Its a solution to prevent one of the SPOFs (Single Point of Failure) i.e. DISK failure. By configuring RAID you are just providing an emergency substitute for the Primary disk. It NEVER means that you have configured DATA backup. | LVM is a disk management approach that allows us to create, extend, reduce, delete or resize the volume groups or logical volumes. |

**By Gopika V Nair**