UNIT –V

Malicious software

Malicious software, or malware is one of the most significant categories of threats to computer systems.

Definition :-

malware is a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim.

Types of Malware (classification) :-

Malware is classified into two broad categories :-

1. how it spreads or propagates to reach the desired targets;
2. the actions or payloads it performs once a target is reached.

Propagation mechanisms include:

- Infection of existing content by viruses that is subsequently spread to other systems
- Exploit of software vulnerabilities by worms or drive-by-downloads to allow the malware to replicate
- Social engineering attacks that convince users to bypass security mechanisms to install Trojans or to respond to phishing attacks

Payload actions performed by malware once it reaches a target system can include:

- Corruption of system or data files
- Theft of service/make the system a zombie agent of attack as part of a botnet
- Theft of information from the system/keylogging
- Stealthing/hiding its presence on the system

Attack Kits

- Initially the development and deployment of malware required considerable technical skill by software authors
- The development of virus-creation toolkits in the early 1990s and then more general attack kits in the 2000s greatly assisted in the development and deployment of malware
- Toolkits are often known as "crimeware"
- Include a variety of propagation mechanisms and payload modules that even novices can deploy
- Variants that can be generated by attackers using these toolkits creates a significant problem for those defending systems against them

- Examples are:
- Zeus
- Angler

Advanced Persistent Threats (APTs) :-

APTs is well-resourced, persistent application of a wide variety of intrusion technologies and malware to selected targets, usually business or political. APTs are typically attributed to state-sponsored organizations, with some attacks likely from criminal enterprises as well.

They are named as a result of these characteristics:

- **Advanced**: Use by the attackers of a wide variety of intrusion technologies and malware, including the development of custom malware if required. The individual components may not necessarily be technically advanced, but are carefully selected to suit the chosen target.
- **Persistent**: Determined application of the attacks over an extended period against the chosen target in order to maximize the chance of success. A variety of attacks may be progressively, and often stealthily, applied until the target is compromised.
- **Threats**: Threats to the selected targets as a result of the organized, capable, and well-funded attackers intent to compromise the specifically chosen targets. The active involvement of people in the process greatly raises the threat level from that due to automated attacks tools, and also the likelihood of successful attack.

The aim of these attacks varies from theft of intellectual property or security and infrastructure related data to the physical disruption of infrastructure. Techniques used include social engineering, spear-phishing emails, and drive-by-downloads from selected compromised websites likely to be visited by personnel in the target organization. The intent is to infect the target with sophisticated malware with multiple propagation mechanisms and payloads. Once they have gained initial access to systems in the target organization, a further range of attack tools are used to maintain and extend their access. As a result, these attacks are much harder to defend against due to this specific targeting and persistence.

Propagation infected contents- viruses:-

The Nature of Viruses

A computer virus is a piece of software that can "infect" other programs, or indeed any type of executable content, by modifying them. The modification includes injecting the original code with a routine to make copies of the virus code, which can then go on to infect other content. The typical virus becomes embedded in a program, or carrier of executable content, on a computer. Then, whenever the infected computer comes into contact with an uninfected piece of code, a fresh copy of the virus passes into the new location

A virus that attaches to an executable program can do anything that the program is permitted to do. It executes secretly when the host program is run. Once the virus code is executing, it can perform any function, such as erasing files and programs that is allowed by the privileges of the current user. One reason viruses dominated the malware scene in earlier years was the lack of user authentication and access controls on personal computer systems at that time. This enabled a virus to infect any executable content on the system. The inclusion of tighter access controls on modern operating systems significantly hinders the ease of infection of such traditional, machine executable code, viruses. This resulted in the development of macro viruses that exploit the active content supported by some documents types, such as Microsoft Word or Excel files, or Adobe PDF documents.

A computer virus has three parts. More generally, many contemporary types of malware also include one or more variants of each of these components:

- **Infection mechanism**: The means by which a virus spreads or propagates, enabling it to replicate. The mechanism is also referred to as the infection vector.
- **Trigger**: The event or condition that determines when the payload is activated or delivered, sometimes known as a logic bomb.
- **Payload**: What the virus does, besides spreading. The payload may involve damage or may involve benign but noticeable activity.

During its lifetime, a typical virus goes through the following four phases:

- **Dormant phase:** The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
- **Propagation phase**: The virus places a copy of itself into other programs or into certain system areas on the disk. The copy may not be identical to the propagating version; viruses often morph to evade detection. Each infected program will now contain a

clone of the virus, which will itself enter a propagation phase.

- **Triggering phase**: The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself.
- **Execution phase:** The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

Most viruses that infect executable program files carry out their work in a manner that is specific to a particular operating system and, in some cases, specific to a particular hardware platform. Thus, they are designed to take advantage of the details and weaknesses of particular systems. Macro viruses though, target specific document types, which are often supported on a variety of systems.

Viruses Classification

As effective countermeasures are developed for existing types of viruses, newer types are developed. There is no simple or universally agreed upon classification scheme for viruses.

A virus classification by target includes the following categories:

- **Boot sector infector**: Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.
- **File infector**: Infects files that the operating system or shell consider to be executable.
- **Macro virus**: Infects files with macro or scripting code that is interpreted by an application.
- **Multipartite virus**: Infects files in multiple ways. Typically, the multipartite virus is capable of infecting multiple types of files, so that virus eradication must deal with all of the possible sites of infection.

A virus classification by concealment strategy includes the following categories:

- **Encrypted virus**: A form of virus that uses encryption to obscure it's content. A portion of the virus creates a random encryption key and encrypts the remainder of the virus. The key is stored with the virus. When an infected program is invoked, the virus uses the stored random key to decrypt the virus. When the virus replicates, a different random key is selected.
- **Stealth virus**: A form of virus explicitly designed to hide itself from detection by anti-virus software. Thus, the entire virus, not just a payload is hidden. It may use code mutation, compression, or rootkit techniques to achieve this.
- **Polymorphic virus**: A form of virus that creates copies during

replication that are functionally equivalent but have distinctly different bit patterns, Propagation—Infected Content—Viruses 209 to defeat programs that scan for viruses. In this case, the "signature" of the virus will vary with each copy. To achieve this variation, the virus may randomly insert superfluous instructions or interchange the order of independent instructions. A more effective approach is to use encryption. The strategy of the encryption virus is followed. The portion of the virus that is responsible for generating keys and performing encryption/decryption is referred to as the mutation engine. The mutation engine itself is altered with each use.

- **Metamorphic virus**: As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, using multiple transformation techniques, increasing the difficulty of detection. Metamorphic viruses may change their behavior as well as their appearance.

Macro and Scripting Viruses

Macro viruses infect scripting code used to support active content in a variety of user document types.

Macro viruses are particularly threatening for a number of reasons:

1. A macro virus is platform independent. Any hardware platform and operating system that supports these applications can be infected.
2. Macro viruses infect documents, not executable portions of code. Most of the information introduced onto a computer system is in the form of documents rather than programs.
3. Macro viruses are easily spread, as the documents they exploit are shared in normal use. A very common method is by electronic mail.

Because macro viruses infect user documents rather than system programs, traditional file system access controls are of limited use in preventing their spread, since users are expected to modify them.

PROPAGATION - VULNARABILITY EXPLOIT WORM

A worm is a program that actively seeks out more machines to infect, and then each infected machine serves as an automated launching pad for attacks on other machines. Worm programs exploit software Vulnerabilities in client or server programs to gain access to each new system. They can use network connections to spread from system to system. They can also spread through shared media, such as USB drives or CD and DVD data disks. E- mail worms spread in macro or script code included in documents attached to e-mail or to instant messenger file transfers. Upon activation, the worm may replicate and propagate again.

To replicate itself, a worm uses some means to access remote systems. These include the following, most of which are still seen in active use

- **Electronic mail or instant messenger facility:** A worm e-mails a copy of itself to other systems, or sends itself as an attachment via an instant message service, so that its code is run when the e- mail or attachment is received or viewed.
- **File sharing:** A worm either creates a copy of itself or infects other suitable files as a virus on removable media such as a USB drive; it then executes when the drive is connected to another system using the autorun mechanism by exploiting some software vulnerability, or when a user opens the infected file on the target system.
- **Remote execution capability**: A worm executes a copy of itself on another system, either by using an explicit remote execution facility or by exploiting a program flaw in a network service to subvert its operations
- **Remote file access or transfer capability:** A worm uses a remote file access or transfer service to another system to copy itself from one system to the other, where users on that system may then execute it.
- **Remote login capability:** A worm logs onto a remote system as a user and then uses commands to copy itself from one system to the other, where it then executes. The new copy of the worm program is then run on the remote system where, in addition to any payload functions that it performs on that system, it continues to propagate.

The propagation phase generally performs the following functions:

- Search for appropriate access mechanisms to other systems to infect by examining host tables, address books, buddy lists, trusted peers, and other similar repositories of remote system access details; by scanning possible target host addresses; or by searching for suitable removable media devices to use.
- Use the access mechanisms found to transfer a copy of itself to the remote system, and cause the copy to be run.

Target Discovery

The first function in the propagation phase for a network worm is for it to search for other systems to infect, a process known as scanning or fingerprinting. For such worms, which exploit software vulnerabilities in remotely accessible network services, it must identify potential systems running the vulnerable service, and then infect them. Then, typically, the worm code now installed on the infected machines repeats the same scanning process, until a large distributed network of infected machines is created.

following types of network address scanning strategies that a worm can use:

• **Random**: Each compromised host probes random addresses in the IP address space, using a different seed. This technique produces a high volume of Internet traffic, which may cause generalized disruption even before the actual attack is launched.

• **Hit-List**: The attacker first compiles a long list of potential vulnerable machines. This can be a slow process done over a long period to avoid detection that an attack is underway. Once the list is compiled, the attacker begins infecting machines on the list. Each infected machine is provided with a portion of the list to scan. This strategy results in a very short scanning period, which may make it difficult to detect that infection is taking place.

• **Topological**: This method uses information contained on an infected victim machine to find more hosts to scan.

• **Local subnet**: If a host can be infected behind a firewall, that host then looks for targets in its own local network. The host uses the subnet address structure to find other hosts that would otherwise be protected by the firewall.

State of Worm Technology

The state of the art in worm technology includes the following:

• **Multiplatform**: Newer worms are not limited to Windows machines but can attack a variety of platforms, especially the popular varieties of UNIX; or exploit macro or scripting languages supported in popular document types.

• **Multi-exploit**: New worms penetrate systems in a variety of ways, using exploits against Web servers, browsers, e-mail, file sharing, and other network-based applications; or via shared media.

• **Ultrafast spreading**: Exploit various techniques to optimize the rate of spread of a worm to maximize its likelihood of locating as many vulnerable machines as possible in a short time period.

• **Polymorphic**: To evade detection, skip past filters, and foil real-time analysis, worms adopt virus polymorphic techniques. Each copy of the worm has new code generated on the fly using functionally equivalent

instructions and encryption techniques.

- **Metamorphic**: In addition to changing their appearance, metamorphic worms have a repertoire of behavior patterns that are unleashed at different stages of propagation.
- **Transport vehicles**: Because worms can rapidly compromise a large number of systems, they are ideal for spreading a wide variety of malicious payloads, such as distributed denial-of-service bots, rootkits, spam e-mail generators, and spyware.
- **Zero-day exploit**: To achieve maximum surprise and distribution, a worm should exploit an unknown vulnerability that is only discovered by the general network community when the worm is launched.

Mobile Code

Mobile code refers to programs (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics. Mobile code is transmitted from a remote system to a local system and then executed on the local system without the user's explicit instruction. Mobile code often acts as a mechanism for a virus, worm, or Trojan horse to be transmitted to the user's workstation. In other cases, mobile code takes advantage of vulnerabilities to perform its own exploits, such as unauthorized data access or root compromise. Popular vehicles for mobile code include Java applets, ActiveX, JavaScript, and VBScript. The most common methods of using mobile code for malicious operations on local system are cross-site scripting, interactive and dynamic Web sites, e-mail attachments, and downloads from untrusted sites or of untrusted software.

propagation—social engineering—spam e-Mail, Trojans:-

Spam email is unsolicited and unwanted junk email sent out in bulk to an indiscriminate recipient list. Typically, spam is sent for commercial purposes. It can be sent in massive volume by botnets, networks of infected computers.

A significant portion of spam e-mail content is just advertising, trying to convince the recipient to purchase some product online, such as pharmaceuticals, or used in scams, such as stock scams or money mule job ads.

But spam is also a significant carrier of malware. The e-mail may have an attached document, which, if opened, may exploit a software vulnerability to install malware on the user's system. Or, it may have an attached Trojan horse program or scripting code that, if run, also installs malware on the user's system. Some Trojans avoid the need for user agreement by exploiting a software vulnerability in order to install themselves, as we discuss next. Finally the spam may be used in a

phishing attack, typically directing the user either to a fake Web site that mirrors some legitimate service, such as an online banking site, where it attempts to capture the user's login and password details; or to complete some form with sufficient personal details to allow the attacker to impersonate the user in an identity theft. All of these uses make spam e-mails a significant security concern.

Trojan Horses: -

A Trojan horse is a useful, or apparently useful, program or utility containing hidden code that, when invoked, performs some unwanted or harmful function. Trojan horse programs can be used to accomplish functions indirectly that the attacker could not accomplish directly. For example, to gain access to sensitive, personal information stored in the files of a user, an attacker could create a Trojan horse program that, when executed, scans the user's files for the desired sensitive information and sends a copy of it to the attacker via a Web form or e-mail or text message.

Trojan horses fit into one of three models:

- Continuing to perform the function of the original program and additionally performing a separate malicious activity.
- Continuing to perform the function of the original program but modifying the function to perform malicious activity or to disguise other malicious activity.

- Performing a malicious function that completely replaces the function of the original program.

Some Trojans avoid the requirement for user assistance by exploiting some software vulnerability to enable their automatic installation and execution. In this they share some features of a worm, but unlike it, they do not replicate.

Mobile Phone Trojans

Trojans Mobile phone Trojans also first appeared in 2004 with the discovery of Skuller. As with mobile worms, the target is the smartphone, and the early mobile Trojans targeted Symbian phones. More recently, a significant number of Trojans have been detected that target Android phones and Apple iPhones. These Trojans are usually distributed via one or more of the app marketplaces for the target phone O/S.

They reviewed over 1200 malware samples found in various Android marketplaces, and noted that 90% of these resulted in the compromised phone being added to a botnet, often with support for accessing premium services or for harvesting user information. They further noted that none of the mobile anti-virus products they tested

were able to detect all of these families. Hence, further development of these products was clearly needed, especially given the rapid evolution of this category of malware.

PAYLOAD –SYSTEM COURRPTION:-

Once malware is active on the target system, the next concern is what actions it will take on this system. That is, what payload does it carry. Some malware has a nonexistent or nonfunctional payload. Its only purpose, either deliberate or due to accidental early release, is to spread. More commonly, it carries one or more payloads that perform covert actions for the attacker. A related payload is one that displays unwanted messages or content on the user's system when triggered.

Data Destruction

The Chernobyl virus is an early example of a destructive parasitic memory- resident Windows-95 and 98 virus that was first seen in 1998. It infects executable files when they are opened. And when a trigger date is reached, it deletes data on the infected system by overwriting the first megabyte of the hard drive with zeroes, resulting in massive corruption of the entire file system. As an alternative to just destroying data, some malware encrypts the user's data, and demands payment in order to access the key needed to recover this information. This is sometimes known as ransom ware. The Gpcode Trojan, which has used public-key cryptography with increasingly larger key sizes to encrypt data. The user needed to pay a ransom, or to make a purchase from certain sites, in order to receive the key to decrypt this data.

Real-World Damage

A further variant of system corruption payloads aims to cause damage to physical equipment. The infected system is clearly the device most easily targeted. The Chernobyl virus mentioned above not only corrupts data, but attempts to rewrite the BIOS code used to initially boot the computer. If it is successful, the boot process fails, and the system is unusable until the BIOS chip is either re-programmed or replaced. More recently, the Stuxnet worm that we discussed previously targets some specific industrial control system software as its key payload.

The centrifuges used in the Iranian uranium enrichment program were strongly suspected as the target, with reports of much higher than normal failure rates observed in them over the period when this worm was active. As noted in our earlier discussion, this has raised concerns over the use of sophisticated targeted malware for industrial sabotage.

Logic Bomb

A key component of data-corrupting malware is the logic bomb. The logic bomb is code embedded in the malware that is set to "explode" when certain conditions are met. Examples of conditions that can be used as triggers for a logic bomb are the presence or absence of certain files or devices on the system, a particular day of the week or date, a particular version or configuration of some software, or a particular user running the application. Once triggered, a bomb may alter or delete data or entire files, cause a machine halt, or do some other damage.

Payload—Attack agent—Zombie, bots :-

The bot is typically planted on hundreds or thousands of computers belonging to unsuspecting third parties. The collection of bots often is capable of acting in a coordinated manner; such a collection is referred to as a botnet. This type of payload attacks the integrity and availability of the infected system.

Uses of Bots lists the following uses of bots:

.Distributed denial-of-service (DDoS) attacks: A DDoS attack is an attack on a computer system or network that causes a loss of service to users.

.Spamming: With the help of a botnet and thousands of bots, an attacker is able to send massive amounts of bulk e-mail (spam).

.Sniffing traffic: Bots can also use a packet sniffer to watch for interesting cleartext data passing by a compromised machine. The sniffers are mostly used to retrieve sensitive information like usernames and passwords.

.Keylogging: If the compromised machine uses encrypted communication channels (e.g. HTTPS or POP3S), then just sniffing the network packets on the victim's computer is useless because the appropriate key to decrypt the packets is missing. But by using a keylogger, which captures keystrokes on the infected machine, an attacker can retrieve sensitive information.

.Spreading new malware: Botnets are used to spread new bots. This is very easy since all bots implement mechanisms to download and execute a file via HTTP or FTP. A botnet with 10,000 hosts that acts as the start base for a worm or mail virus allows very fast spreading and thus causes more harm.

.Installing advertisement add-ons and browser helper objects (BHOs): Botnets can also be used to gain financial advantages. This works by setting up a fake Web site with some advertisements: The

operator of this Web site negotiates a deal with some hosting companies that pay for clicks on ads. With the help of a botnet, these clicks can be "automated" so that instantly a few thousand bots click on the pop-ups. This process can be further enhanced if the bot hijacks the start-page of a compromised machine so that the "clicks" are executed each time the victim uses the browser.

•**Attacking IRC chat networks**: Botnets are also used for attacks against Internet Relay Chat (IRC) networks. Popular among attackers is especially the so called clone attack: In this kind of attack, the controller orders each bot to connect a large number of clones to the victim IRC network.

•**Manipulating online polls/games**: Online polls/games are getting more and more attention and it is rather easy to manipulate them with botnets. Since every bot has a distinct IP address, every vote will have the same credibility as a vote cast by a real person. Online games can be manipulated in a similar way.

Payload– Information Theft– Keyloggers, Phishing, Spyware: - Keyloggers

Credential Theft, Keyloggers, and Spyware Typically, users send their login and password credentials to banking, gaming, and related sites over encrypted communication channels. which protects them from capture by monitoring network packets. To bypass this, an attacker can install a keylogger, which captures keystrokes on the infected machine to allow an attacker to monitor this sensitive information. Since this would result in the attacker receiving a copy of all text entered on the compromised machine, keyloggers typically implement some form of filtering mechanism that only returns information close to desired keywords (e.g., "login" or "password" or "paypal.com").

In response to the use of keyloggers, some banking and other sites switched to using a graphical applet to enter critical information, such as passwords. Since these do not use text entered via the keyboard, traditional keyloggers do not capture this information. In response, attackers developed more general spyware payloads, which subvert the compromised machine to allow monitoring of a wide range of activity on the system. This may include monitoring the history and content of browsing activity, redirecting certain Web page requests to fake sites controlled by the attacker, and dynamically modifying data exchanged between the browser and certain Web sites of interest. It steals banking and financial credentials using both a keylogger and capturing and possibly altering form data for certain Web sites. It is typically deployed using either spam e-mails or via a compromised Web site in a "drive-by-download."

Phishing and Identity Theft

Another approach used to capture a user's login and password credentials is to include a URL in a spam e-mail that links to a fake Web site controlled by the attacker, but which mimics the login page of some banking, gaming, or similar site. This is normally included in some message suggesting that urgent action is required by the user to authenticate their account, to prevent it being locked. If the user is careless, and does not realize that they are being conned, then following the link and supplying the requested details will certainly result in the attackers exploiting their account using the captured credentials.

Such a spam e-mail may direct a user to a fake Web site Controlled by the attacker, or to complete some enclosed form and return to an e-mail accessible to the attacker, which is used to gather a range of private, personal, information on the user. Given sufficient details, the attacker can then "assume" the user's identity for the purpose of obtaining credit, or sensitive access to other resources. This is known as a phishing attack and exploits social engineering to leverage user's trust by masquerading as communications from a trusted source. Such general spam e-mails are typically widely distributed to very large numbers of users, often via a botnet. While the content will not match appropriate trusted sources for a significant fraction of the recipients, the attackers rely on it reaching sufficient users of the named trusted source, a gullible portion of whom will respond, for it to be profitable. A more dangerous variant of this is the spear-phishing attack. This again is an e-mail claiming to be from a trusted source. However, the recipients are carefully researched by the attacker, and each e-mail is carefully crafted to suit its recipient

Reconnaissance, Espionage and Data Exfiltration Credential theft and identity theft are special cases of a more general reconnaissance payload, which aims to obtain certain types of desired information and return this to the attacker. These special cases are certainly the most common; however, other targets are known. Operation Aurora in 2009 used a Trojan to gain access to and potentially modify source code repositories at a range of high tech, security, and defense contractor companies. The Stuxnet worm discovered in 2010 included capture of hardware and software configuration details in order to determine whether it had compromised the specific desired target systems. Early versions of this worm returned this same information, which was then used to develop the attacks deployed in later versions. APT attacks may result in the loss of large volumes of sensitive information, which is sent, exfiltrated from the target organization, to the attackers. To detect and block such data exfiltration requires suitable "data-loss"

technical countermeasures that manage either access to such information, or its transmission across the organization's network perimeter.

Backdoor A backdoor, also known as a trapdoor, is a secret entry point into a program that allows someone who is aware of the backdoor to gain access without going through the usual security access procedures. Programmers have used backdoors legitimately for many years to debug and test programs; such a backdoor is called a maintenance hook. This usually is done when the programmer is developing an application that has an authentication procedure, or a long setup, requiring the user to enter many different values to run the application. To debug the program, the developer may wish to gain special privileges or to avoid all the necessary setup and authentication. The programmer may also want to ensure that there is a method of activating the program should something be wrong with the authentication procedure that is being built into the application. The backdoor is code that recognizes some special sequence of input or is triggered by being run from a certain user ID or by an unlikely sequence of events. Backdoors become threats when unscrupulous programmers use them to gain unauthorized access. The backdoor was the basic idea for the vulnerability. It is difficult to implement operating system controls for backdoors in applications. Security measures must focus on the program development and software update activities, and on programs that wish to offer a network service.

Rootkit A rootkit is a set of programs installed on a system to maintain covert access to that system with administrator (or root)3 privileges, while hiding evidence of its presence to the greatest extent possible. This provides access to all the functions and services of the operating system. The rootkit alters the host's standard functionality in a malicious and stealthy way. With root access, an attacker has complete control of the system and can add or change programs and files, monitor processes, send and receive network traffic, and get backdoor access on demand. A rootkit can make many changes to a system to hide its existence, making it difficult for the user to determine that the rootkit is present and to identify what changes have been made. In essence, a rootkit hides by subverting the mechanisms that monitor and report on the processes, files, and registries on a computer. A rootkit can be classified using the following characteristics:

- Persistent: Activates each time the system boots. The rootkit must store code in a persistent store, such as the registry or file system, and configure a method by which the code executes without user intervention. This means it is easier to detect, as the copy in

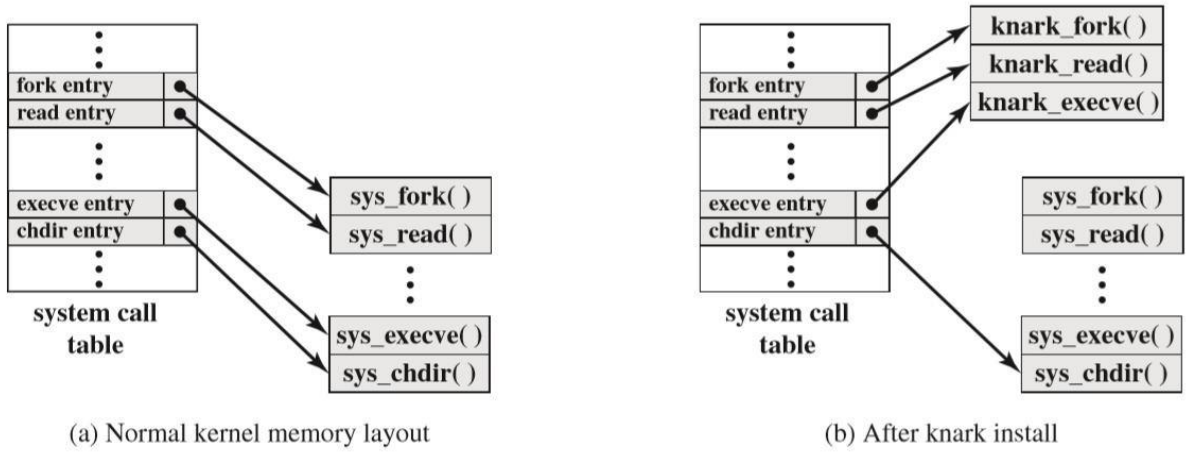
persistent storage can potentially be scanned.

- •Memory based: Has no persistent code and therefore cannot survive a reboot. However, because it is only in memory, it can be harder to detect.
- User mode: Intercepts calls to APIs (application program interfaces) and modifies returned results. For example, when an application performs a directory listing, the return results do not include entries identifying the files associated with the rootkit.
- Kernel mode: Can intercept calls to native APIs in kernel mode.The rootkit can also hide the presence of a malware process by removing it from the Kernel's list of active processes.
- Virtual machine based: This type of rootkit installs a lightweight virtual machine monitor, and then runs the operating system in a virtual machine The rootkit can then transparently intercept and modify states and events occurring in the virtualized system.
- External mode: The malware is located outside the normal operation mode of the targeted system, in BIOS or system management mode, where it can directly access hardware.

Kernel Mode Rootkits The next generation of rootkits moved down a layer, making changes inside the kernel and co-existing with the operating systems code, in order to make their detection much harder. Any "anti-virus" program would now be subject to the same "low-level" modifications that the rootkit uses to hide its presence. However, methods were developed to detect these changes.

three techniques that can be used to change system calls:

- Modify the system call table: The attacker modifies selected syscall addresses stored in the system call table. This enables the rootkit to direct a system call away from the legitimate routine to the rootkit's replacement. figure shows how the knark rootkit achieves this.
- Modify system call table targets: The attacker overwrites selected legitimate system call routines with malicious code. The system call table is not changed.
- Redirect the system call table: The attacker redirects references to the entire system call table to a new table in a new kernel memory location.



(a) Normal kernel memory layout (b) After knark install

System Call Table Modification by Rootkit

- **COUNTERMEASURES:-**

We now consider possible countermeasures for malware. These are generally known as "anti-virus" mechanisms, as they were first developed to specifically target virus infections. However, they have evolved to address most of the types of malware.

Malware Countermeasure Approaches The ideal solution to the threat of malware is prevention: Do not allow malware to get into the system in the first place, or block the ability of it to modify the system. This goal is, in general, nearly impossible to achieve, although taking suitable countermeasures to harden systems and users in preventing infection can significantly reduce the number of successful malware attacks. Suggests there are four main elements of prevention: policy, awareness, vulnerability mitigation, and threat mitigation. Having a suitable policy to address malware prevention provides a basis for implementing appropriate preventative countermeasures.

One of the first countermeasures that should be employed is to ensure all systems are as current as possible, with all patches applied, in order to reduce the number of vulnerabilities that might be exploited on the system. The next is to set appropriate access controls on the applications and data stored on the system, to reduce the number of files that any user can access, and hence potentially infect or corrupt, as a result of them executing some malware code. These measures directly target the key propagation mechanisms used by worms, viruses, and some Trojans. The third common propagation mechanism, which targets users in a social engineering attack, can be countered using appropriate user awareness and training. This aims to equip users to be more aware of these attacks, and less likely to take actions that result in their compromise. If prevention fails, then technical mechanisms can be used to support the
following threat mitigation options:

- **Detection**: Once the infection has occurred, determine that it has occurred and locate the malware.

- **Identification**: Once detection has been achieved, identify the specific malware that has infected the system.
- **Removal**: Once the specific malware has been identified, remove all traces of malware virus from all infected systems so that it cannot spread further. If detection succeeds but either identification or removal is not possible, then the alternative is to discard any infected or malicious files and reload a clean backup

version. In the case of some particularly nasty infections, this may require a complete wipe of all storage, and rebuild of the infected system from known clean media.

To begin, let us consider some requirements for effective malware countermeasures: •Generality: The approach taken should be able to handle a wide variety of attacks.

- **Timeliness**: The approach should respond quickly so as to limit the number of infected programs or systems and the consequent activity.
- **Resiliency**: The approach should be resistant to evasion techniques employed by attackers to hide the presence of their malware.
- **Minimal denial-of-service costs**: The approach should result in minimal reduction in capacity or service due to the actions of the countermeasure software, and should not significantly disrupt normal operation.
- **Transparency**: The countermeasure software and devices should not require modification to existing (legacy) OSs, application software, and hardware.
- **Global and local coverage**: The approach should be able to deal with attack sources both from outside and inside the enterprise network.

Host-Based Scanners The first location where anti-virus software is used is on each end system. This gives the software the maximum access to information on not only the behavior of the malware as it interacts with the targeted system, but also the smallest overall view of malware activity. The use of anti-virus software on personal computers is now widespread, in part caused by the explosive growth in malware volume and activity. This software can be regarded as a form of host-based intrusion detection system, which we discuss more generally in Section 8.4. Advances in virus and other malware technology, and in antivirus technology and other countermeasures, go hand in hand. Early malware used relatively simple and easily detected code, and hence could be identified and purged with relatively simple anti-virus software packages. As the malware arms race has evolved, both the malware code and, necessarily, anti-virus software have grown more complex and sophisticated.

identifies four generations of anti-virus software:

- First generation: simple scanners- A first-generation scanner requires a malware signature to identify the malware. The signature may contain "wildcards" but matches essentially the same structure and bit pattern in all copies of the malware. Such signature-specific scanners are limited to the detection of known malware. Another type of first-generation scanner maintains a record of the length of programs and looks for changes in length as a result of virus infection.
- Second generation:

heuristic scanners -A second-generation scanner does not rely on a specific signature. Rather, the scanner uses heuristic rules to search for probable malware instances. One class of such scanners looks for fragments of code that are often associated with malware.

Another second-generation approach is integrity checking. A checksum can be appended to each program. If malware alters or replaces some program without changing the checksum, then an integrity check will catch this change.

- Third generation: activity traps -Third-generation programs are memory-resident programs that identify malware by its actions rather than its structure in an infected program. Such programs have the advantage that it is not necessary to develop signatures and heuristics for a wide array of malware. Rather, it is necessary only to identify the small set of actions that indicate malicious activity is being attempted and then to intervene.
- Fourth generation: full-featured protection- Fourth-generation products are packages consisting of a variety of anti-virus techniques used in conjunction. These include scanning and activity trap components. In addition, such a package includes access control capability, which limits the ability of malware to penetrate a system and then limits the ability of a malware to update files in order to propagate.

Host-based behavior-blocking software- Unlike heuristics or fingerprint based scanners, behavior-blocking software integrates with the operating system of a host computer and monitors program behavior in real time for malicious actions. It is a type of host-based intrusion prevention system, The behavior blocking software then blocks potentially malicious actions before they have a chance to affect the system. Monitored behaviors can include:

- Attempts to open, view, delete, and/or modify files;
- Attempts to format disk drives and other unrecoverable disk operations;
- Modifications to the logic of executable files or macros;
- Modification of critical system settings, such as start-up settings;
- Scripting of e-mail and instant messaging clients to send executable content; and
- Initiation of network communications.

spyware Detection and removal Although general anti-virus

products include signatures to detect spyware, the threat this type of malware poses, and its use of stealthing techniques, means that a range of spyware specific detection and removal utilities exist. These specialize in the detection and removal of spyware, and provide more robust capabilities. Thus they complement, and should be used along with, more general anti-virus products.

Rootkit countermeasures Rootkits can be extraordinarily difficult to detect and neutralize, particularly so for kernel-level rootkits. Many of the administrative tools that could be used to detect a rootkit or its traces can be compromised by the rootkit precisely so that it is undetectable.