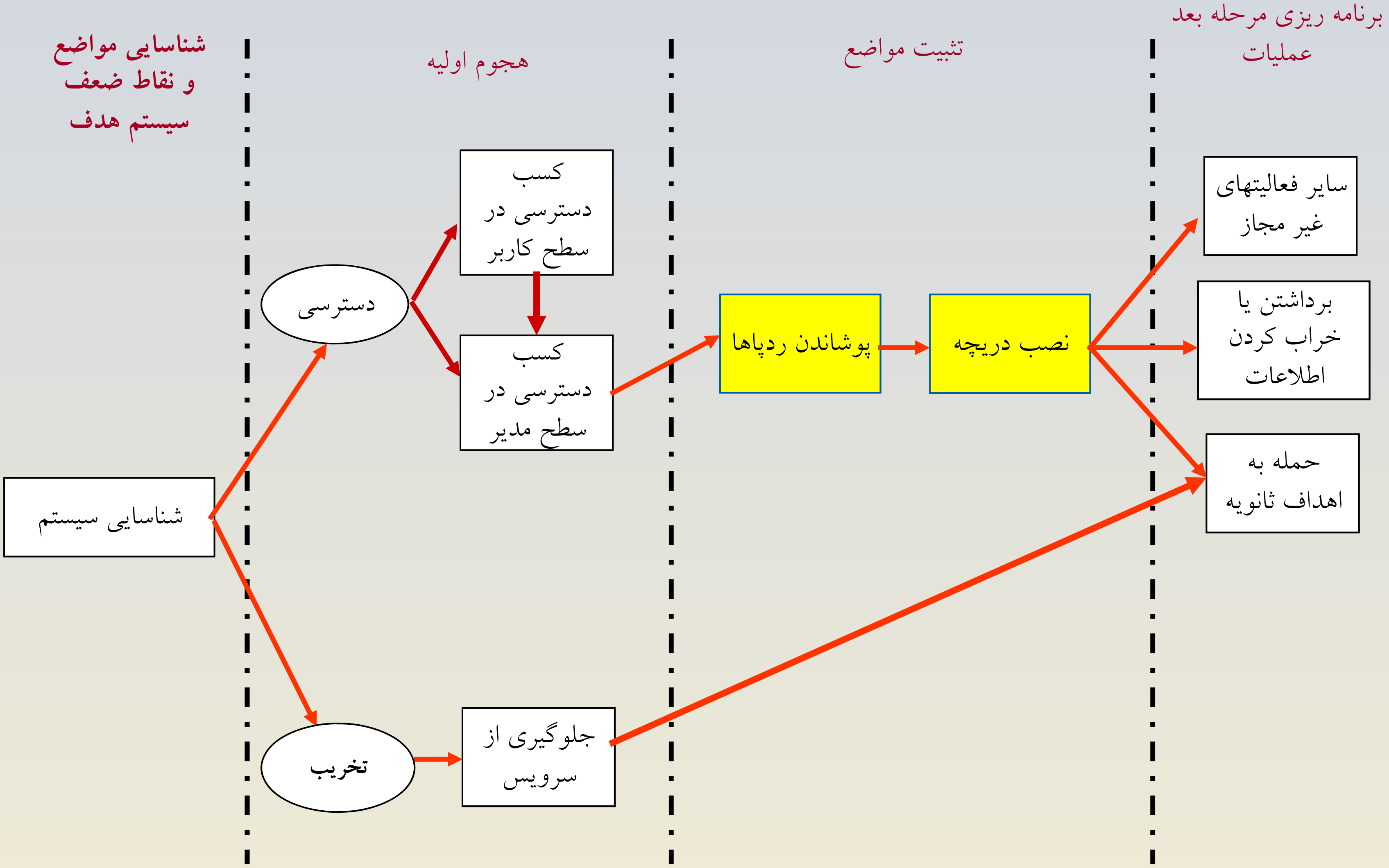
# مروری بر نفوذگری و امنیت در سیستم‌های کامپیوتری

## تثبیت مواضع

# روند نمای کلی انجام یک حملۀ کامپیوتری



# Contents

- Definitions
- Spywares & Trojan horses
- Rootkits
- Covert channels

# Definitions

SPYWARE

A general term for a program that secretly monitors your actions. While they are not sometimes malicious, but like a remote control program used by a hacker receive your private information. Software companies have been known to use Spyware to gather data about customers.

**Definition from:** *BlackICE Internet Security Systems - http://blackice.iss.net/glossary.php*

TROJAN HORSE

An apparently useful and innocent program containing additional hidden code which allows the unauthorized collection, exploitation, falsification, or destruction of data.

**Definition from:** *Texas State Library and Archives Commission - http://www.tsl.state.tx.us/ld/pubs/compsecurity/glossary.html*

# Summary of Effects

- Collection of data from your computer without your agreement
- Collection of data pertaining to your habitual use
- Execution of code without your agreement
- Installation on your computer without your agreement
- Inability to remove the software
- Performing other undesirable tasks without agreement

# Spyware Software Examples

- GAIN / Gator
- Gator E-Wallet
- Cydoor
- BonziBuddy
- Google Toolbar
- Yahoo Toolbar
- DownloadWare
- BrowserAid
- Dogpile Toolbar

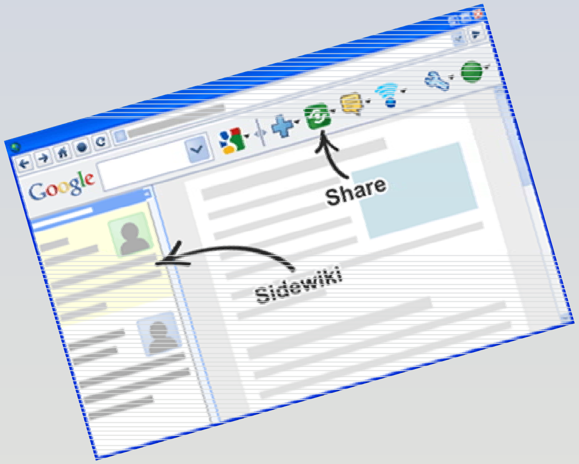


**Image Sources…**

**GAIN Logo –** The Gator Corporation – http://www.gator.com
**BonziBuddy Logo –** Bonzi.com - http://images.bonzi.com/images/gorillatalk.gif
**DownloadWare Logo** – DownloadWare - http://www.downloadware.net

# Spyware Defence

**User Initiatives…**

- Use Legitimate S/W Sources
- Improved Technical Ability
- Choice of Browser
- Choice of OS

**Technical Initiatives...**

- Spyware Removal Programs
- Firewall Technology
- Disable ActiveX Controls
- E-Mail Filters
- Download Patches

# Types of Trojan Horse

- **Remote Access Trojan**: allow attacker to gain control over the victim's pc.
- **Data sending Trojan**: provide the attacker confidential data such as password, credit card information.
- **Destructive Trojan**: designed to destroy or delete files.
- **Proxy Trojan**: to use the victim's computer as the proxy server for the attackers.
- **FTP Trojan**: designed to open ftp port (port 21) on your computer, enable the attacker to connect your PC through File Transfer Protocol.
- **Security software disabler Trojan**: designed to stop or kill security software program such as antivirus program and internet security program.
- **Denial of Service (DoS) attack**: the attacker try to bring down the network service by flooding the useless traffic over the network.

# Solutions

## Use the following security mechanisms

- Firewall
- Virus Checker
- Spyware Remover
- Frequent OS updates
- Frequent back-up
- Learning problems

# Similarities / Differences

| Spyware | Trojan Horses |
|---|---|
| Commercially Motivated | Malicious |
| Internet connection required | Any network connection required |
| Initiates remote connection | Receives incoming connection |
| Purpose: To monitor activity | Purpose: To control activity |
| Collects data | Unauthorized access and control |
| Legal | Illegal |
| Not Detectable with Virus Checker | Detectable with Virus Checker |
| Age: Relatively New (< 10 Years) | Age: Relatively Old ( > 20 Years) |
| Memory Resident Processes | |
| Secretly installed without user’s consent or understanding | |
| Creates a security vulnerability | |

**Source** – Table derived and produced by; Andrew Brown, Tim Cocks and Kumutha Swampillai, February 2004.

# Contents

- Definitions
- Spywares & Trojan horses
- Rootkits
- Covert channels

# What is a Rootkit?

- A rootkit is a tool that is designed to hide itself and other processes, data, and/or activity on a system.
- “A tool used to protect backdoors and other tools from detection by administrators”
- A rootkit is not
  - An exploit
  - A virus or worm

# Rootkits - Why Should You Care?

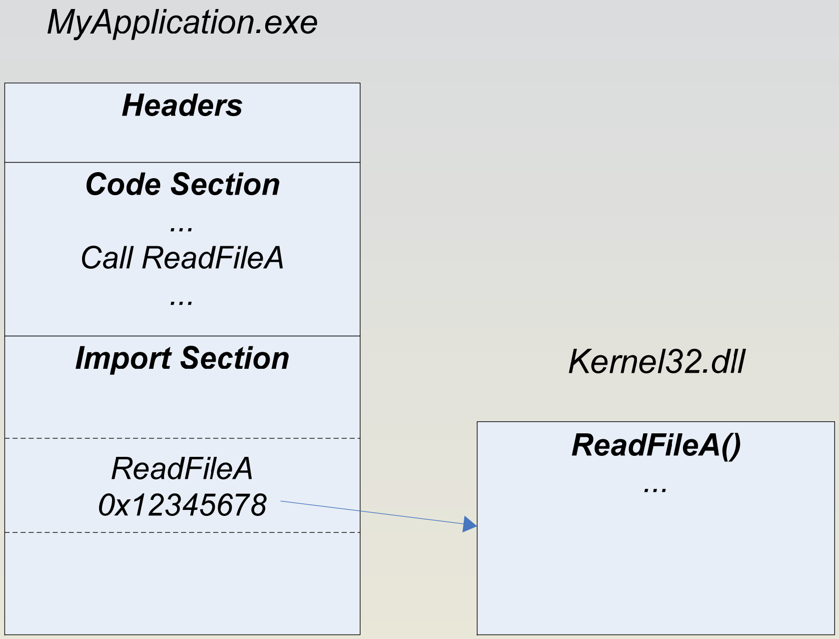
- If you can’t detect a backdoor on any given machine, how do you know your machine is clean?
- New viruses will use new rootkit technology

# Rootkits - How They Work?

- To hide in a system you have to control a system
- Act as a gatekeeper between what a user sees and what the system sees
- Requires administrator privileges to install

# How Rootkits Work - Hooking

- A standard application



*MyApplication.exe*

***Headers***

***Code Section***

*...*

*Call ReadFileA*

*...*

***Import Section***

*ReadFileA*

*0x12345678*

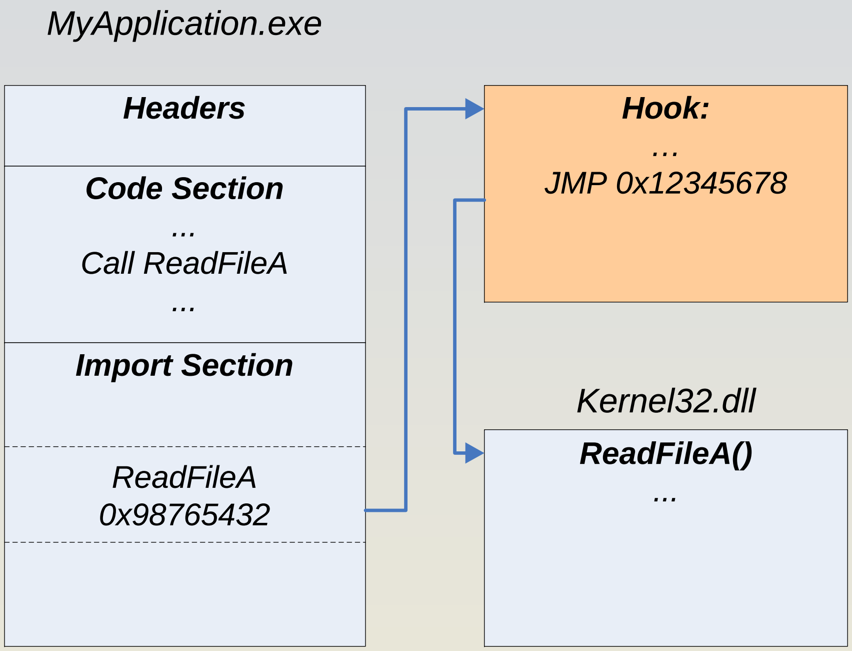
*Kernel32.dll*

***ReadFileA()***

*...*

# How Rootkits Work - Hooking

- **A hooked application**

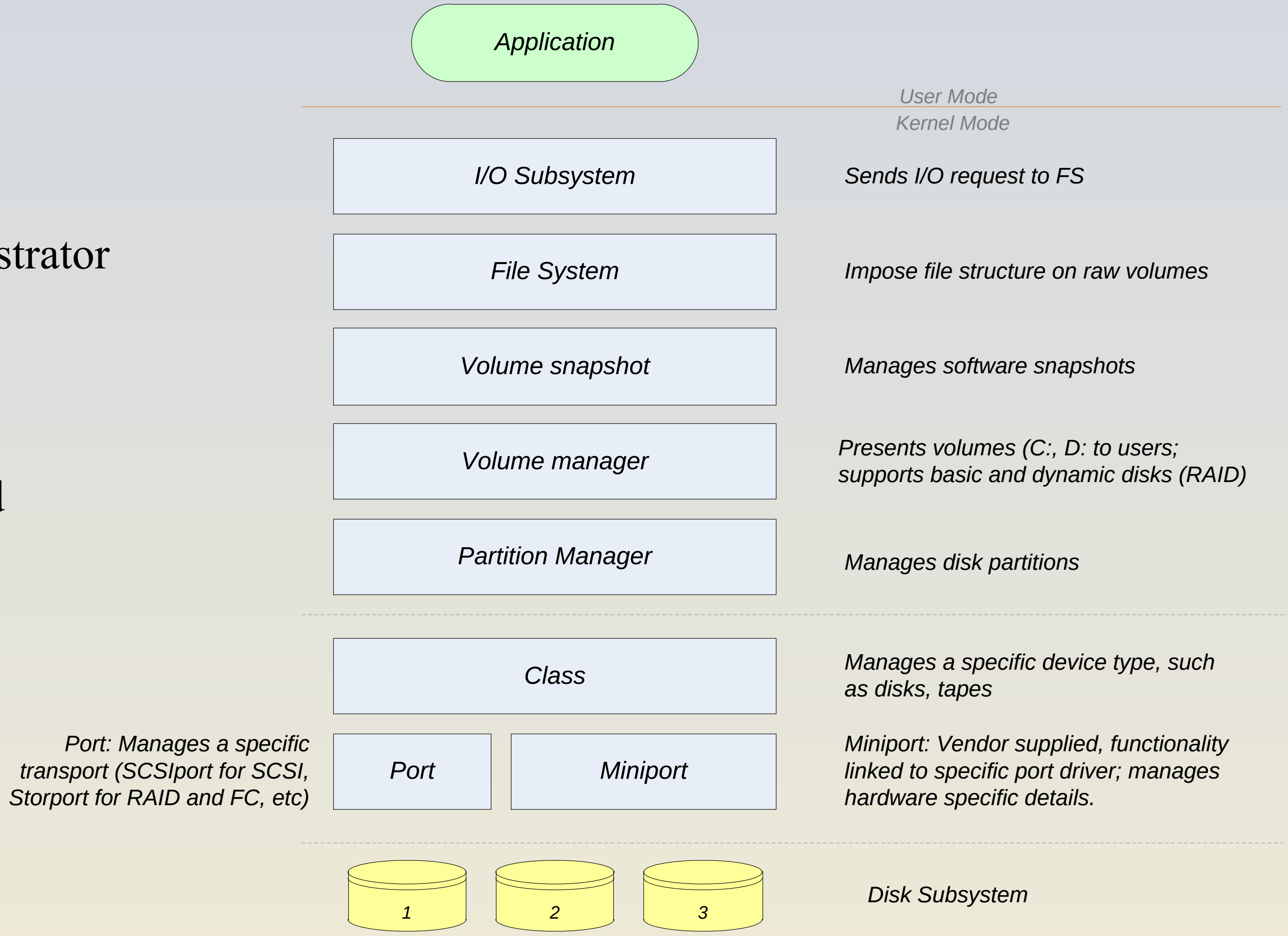


# Rootkits – How They Work?

- To hide what is taking place, an attacker wants to:
  - Hide processes
  - Hide services
  - Hide listening TCP/UDP ports
  - Hide kernel modules
  - Hide drivers

# Levels of Access in Windows

- User Land
  - User
  - Administrator
  - System
- Kernel Land
  - Drivers



Application

User Mode

Kernel Mode

I/O Subsystem — Sends I/O request to FS

File System — Impose file structure on raw volumes

Volume snapshot — Manages software snapshots

Volume manager — Presents volumes (C:, D: to users; supports basic and dynamic disks (RAID)

Partition Manager — Manages disk partitions

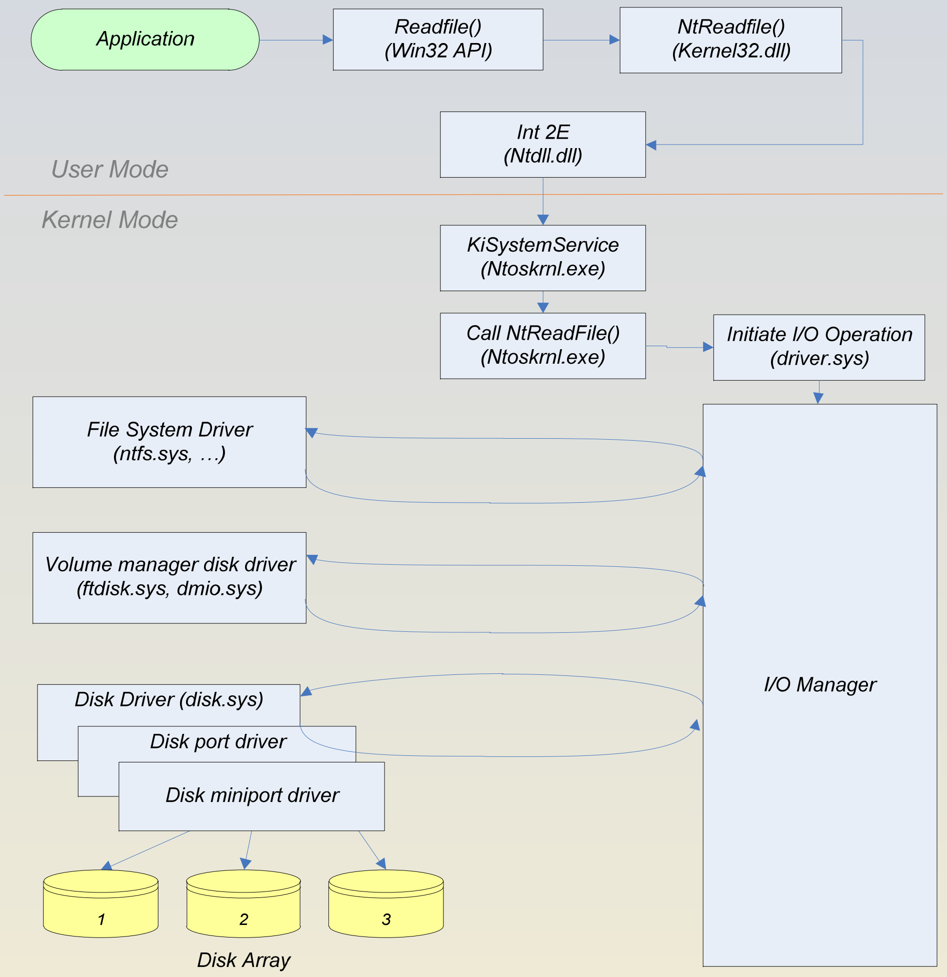
Class — Manages a specific device type, such as disks, tapes

Port — Port: Manages a specific transport (SCSIport for SCSI, Storport for RAID and FC, etc)

Miniport — Miniport: Vendor supplied, functionality linked to specific port driver; manages hardware specific details.

1 2 3 — Disk Subsystem

# What Happens When You Read a File?

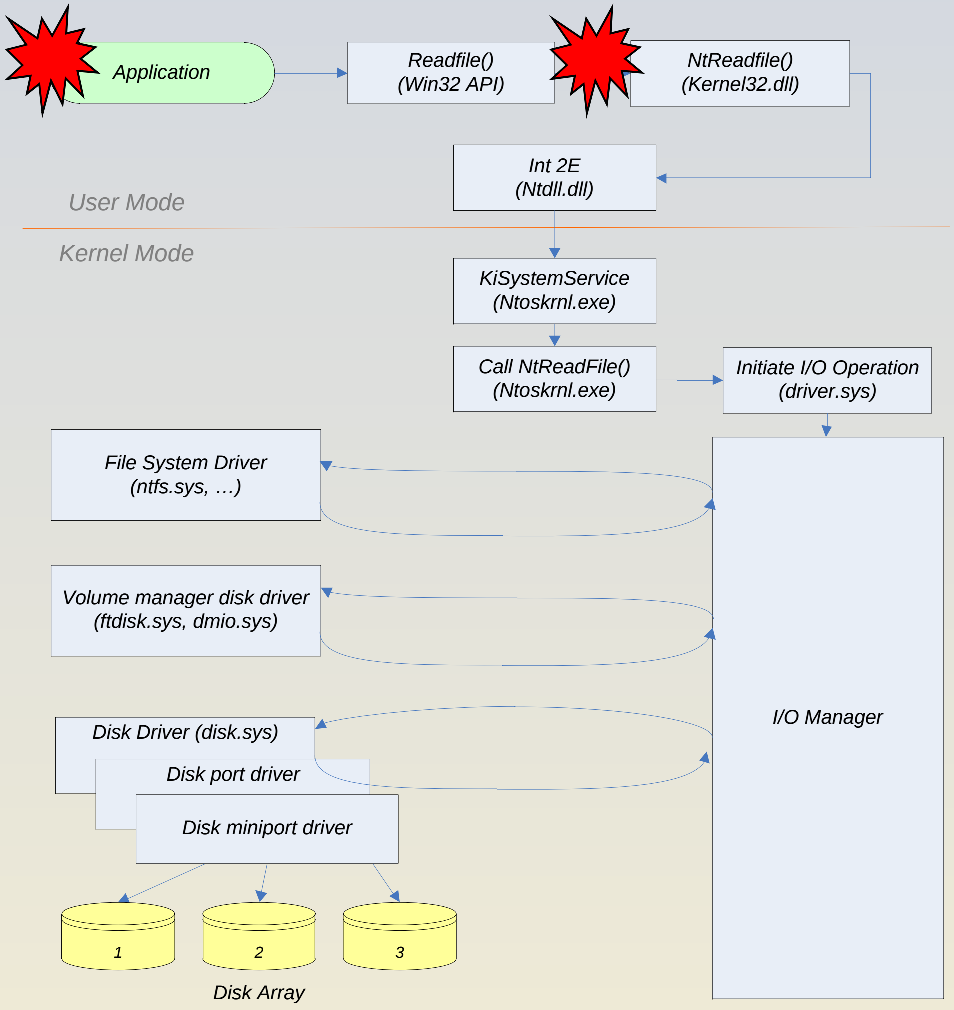


- Readfile() called on File1.txt
- Transition to Ring 0
- NtReadFile() processed
- I/O Subsystem called
- IRP generated

- Data at File1.txt requested from ntfs.sys

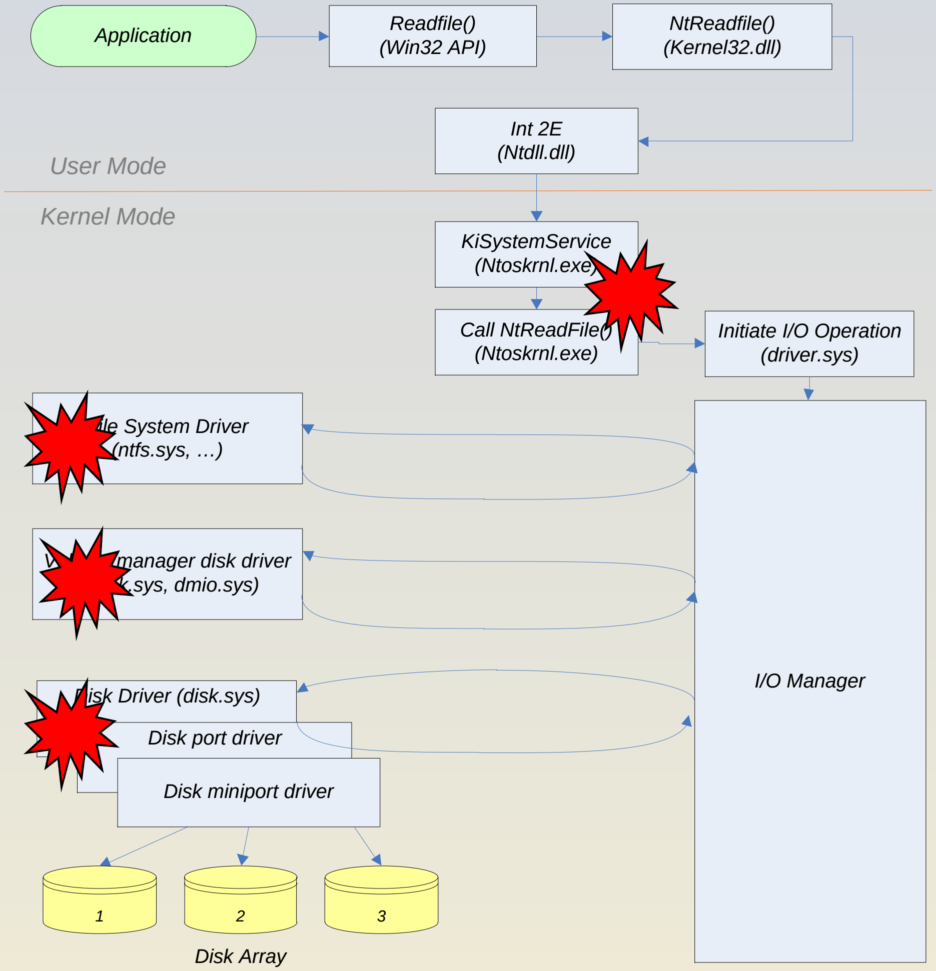
- Data on disk 2 requested from disk.sys

# Userland Rootkits



- Binary replacement eg modified Exe or Dll
- Binary modification in memory eg He4Hook
- User land hooking eg Hacker Defender
  - IAT hooking

# Kernel (Ring 0) Rootkits



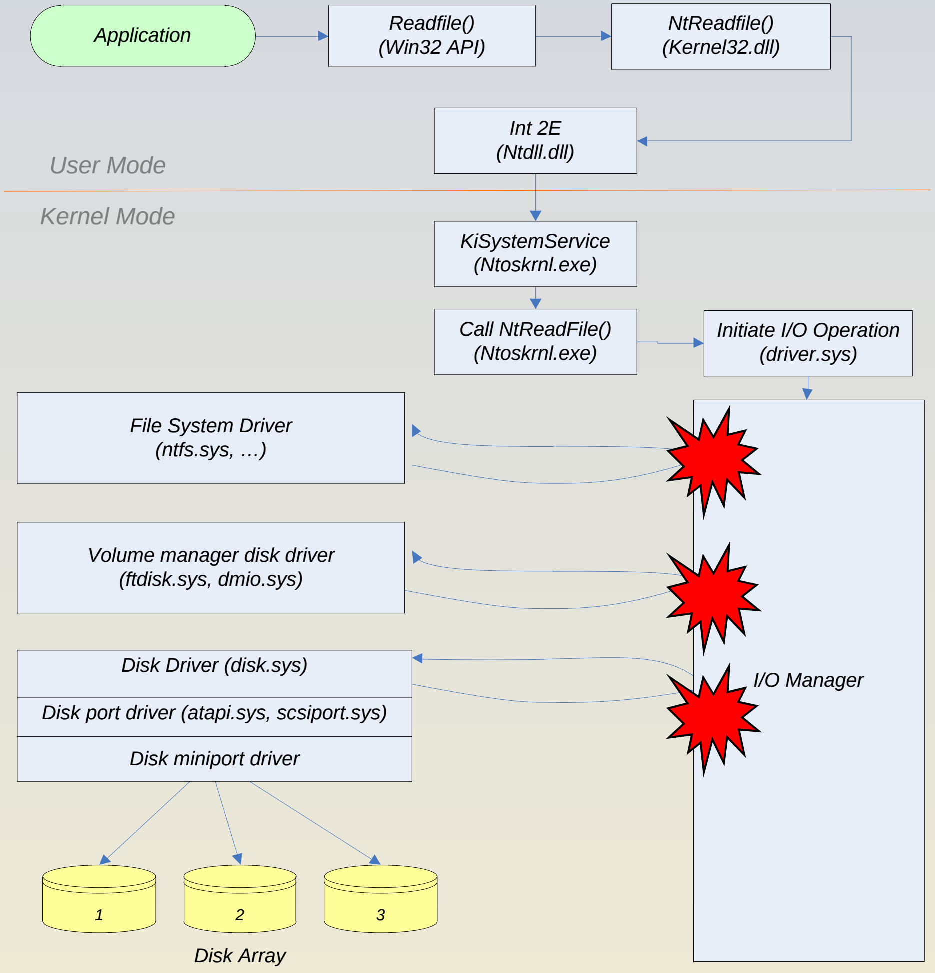
- Kernel Hooking

  E.g. NtRootkit

- Driver replacement

  E.g. replace ntfs.sys with ntfss.sys

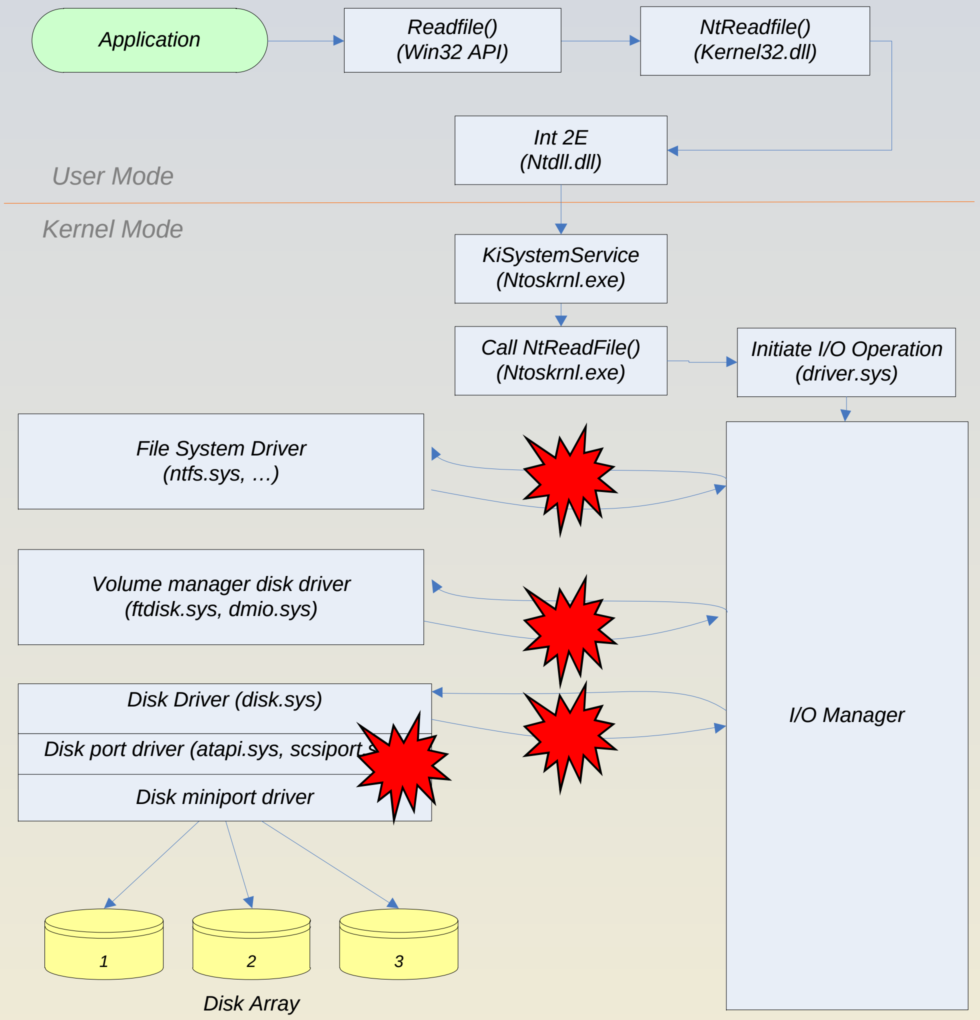
# Kernel (Ring 0) Rootkits



- **IO Request Packet (IRP) Hooking**
  - IRP Dispatch Table

E.g. He4Hook (some versions)

# Kernel (Ring 0) Rootkits



- **Filter Drivers**
- **Types**
  - File system filter
  - Volume filter
  - Disk Filter
  - Bus Filter

# Current Rootkit Capabilities

- Hide processes
- Hide files
- Hide registry entries
- Hide services
- Completely bypass personal firewalls
- Undetectable by anti virus
- Covert channels - undetectable on the network
- Install silently

# Detection Methodologies

- Traditional Detection
  - Check integrity of important OS elements against a hash database (sigcheck)
  - Look for unidentified processes (task manager)
  - Check for open ports (netstat)

# Detection Methodologies

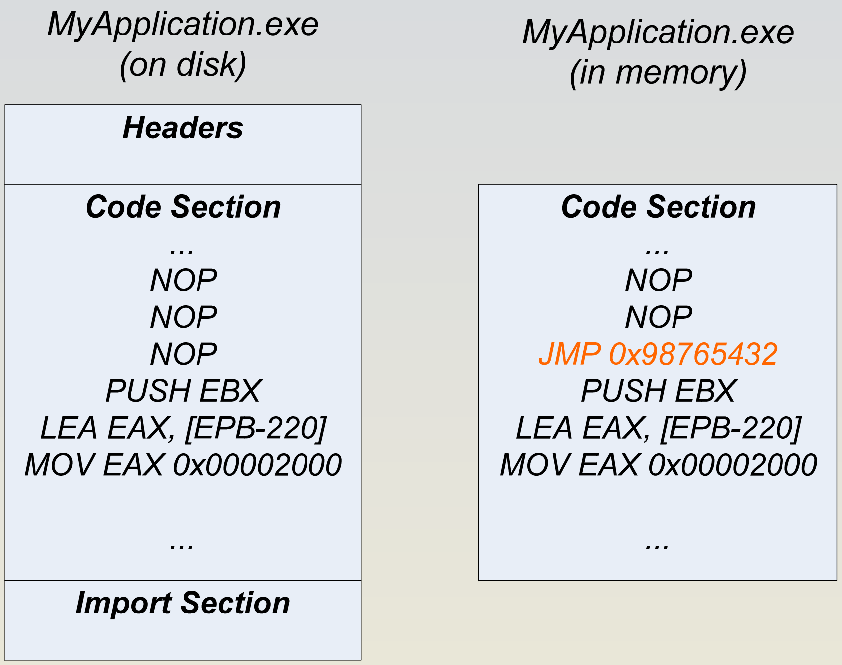
- Signature based
  - Look for known rootkits, viruses, backdoors
  - Antivirus
  - Look for “bad things” living in memory

- Problems
  - Requires updated databases
  - Doesn’t detect anything it hasn’t seen before

# Detection Methodologies

- Code verification
  - Code sections are read only in all modern OSes
  - Programs should not modify their own code
  - Check to see if the files on disk match what is running in memory

# Detection Methodologies: Code Verification



*MyApplication.exe*
*(on disk)*

***Headers***

***Code Section***

*...*
*NOP*
*NOP*
*NOP*
*PUSH EBX*
*LEA EAX, [EPB-220]*
*MOV EAX 0x00002000*

*...*

***Import Section***

*MyApplication.exe*
*(in memory)*

***Code Section***

*...*
*NOP*
*NOP*
*JMP 0x98765432*
*PUSH EBX*
*LEA EAX, [EPB-220]*
*MOV EAX 0x00002000*

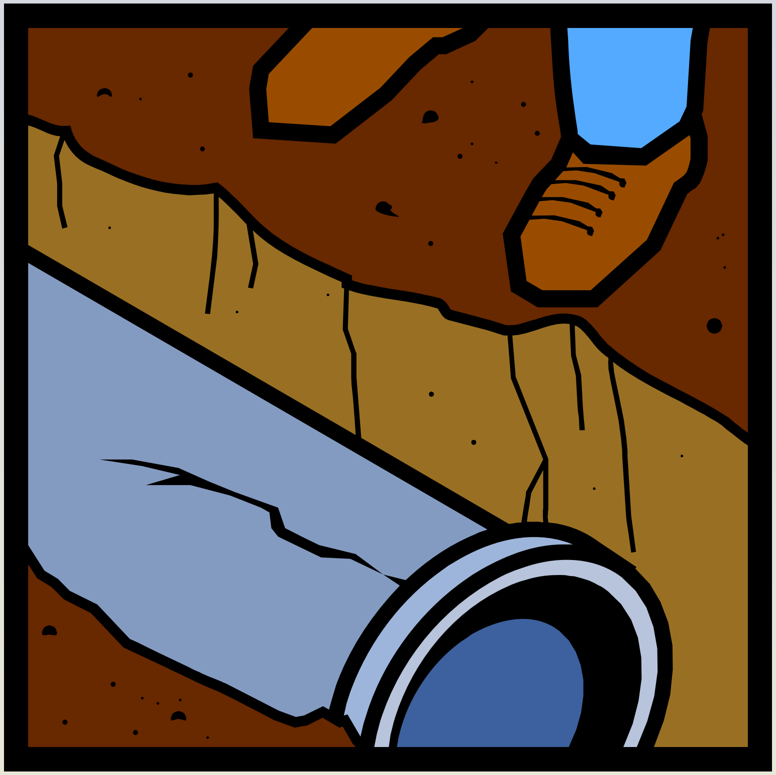
*...*

# Hardware Rootkits

- A OS reinstall won’t save you
- Hard to remove.
  - Device is usually destroyed
- Difficult to implement

# Contents

- Definitions
- Spywares & Trojan horses
- Rootkits
- Covert channels



# Covert Channel

- **Covert channel**: a communication path not intended as such by system's designers

- For example, resources shared at different levels could be used to "signal" information

# Covert Channel Example

- Alice has **TOP SECRET**, she wants to reveal it to Bob
- Suppose the file space shared by all users
- Alice creates file FileXYzW to signal “1” to Bob, and removes file to signal “0”
- Bob lists the files
  - If file FileXYzW does not exist, Alice sent 0
  - If file FileXYzW exists, Alice sent 1
- Alice can leak **TOP SECRET** info to Bob!

# Covert Channel Example

| | | | | | |
|---|---|---|---|---|---|
| **Alice:** | Create file | Delete file | Create file | | Delete file |
| **Bob:** | Check file | Check file | Check file | Check file | Check file |
| **Data:** | 1 | 0 | 1 | 1 | 0 |

**Time:**

# Covert Channel

- Other possible covert channels?
  - Print queue
  - ACK messages
  - Network traffic, etc.
- When does covert channel exist?
  1. Sender and receiver have a shared resource
  2. Sender able to vary some property of resource that receiver can observe
  3. “Communication” between sender and receiver can be synchronized

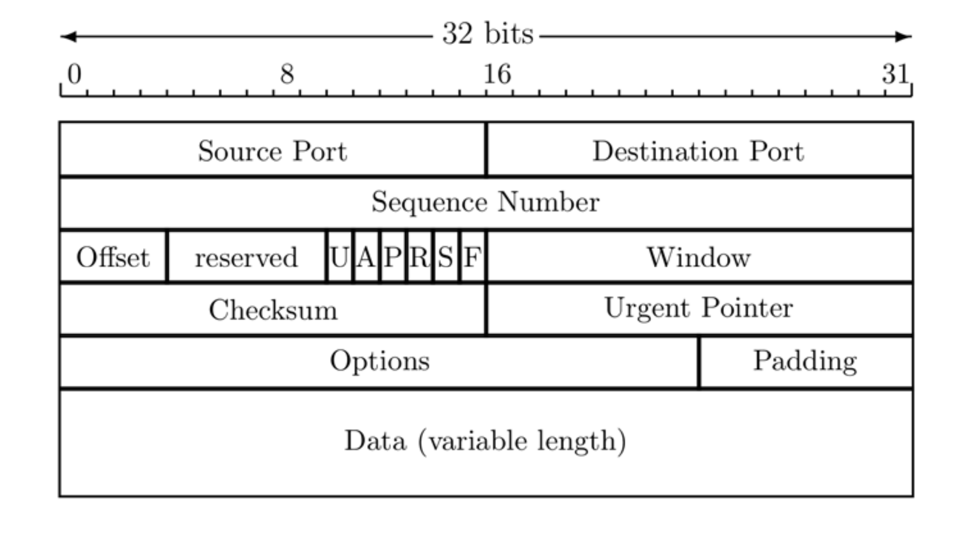
# Covert Channel

- So, covert channels are everywhere
- "Easy" to eliminate covert channels:
  - Eliminate all shared resources…
  - …and all communication
- Virtually impossible to eliminate covert channels in any useful system
  - DoD guidelines: **reduce covert channel capacity** to no more than 1 bit/second
  - Implication? DoD has given up on *eliminating* covert channels!

## Covert Channel

- Consider 100MB **TOP SECRET** file
  - Plaintext stored in **TOP SECRET** location
  - Ciphertext (encrypted with AES using 256-bit key) stored in **UNCLASSIFIED** location
- Suppose we reduce covert channel capacity to 1 bit per second
- It would take more than 25 years to leak entire document thru a covert channel
- But it would take less than 5 minutes to leak 256-bit AES key thru covert channel!

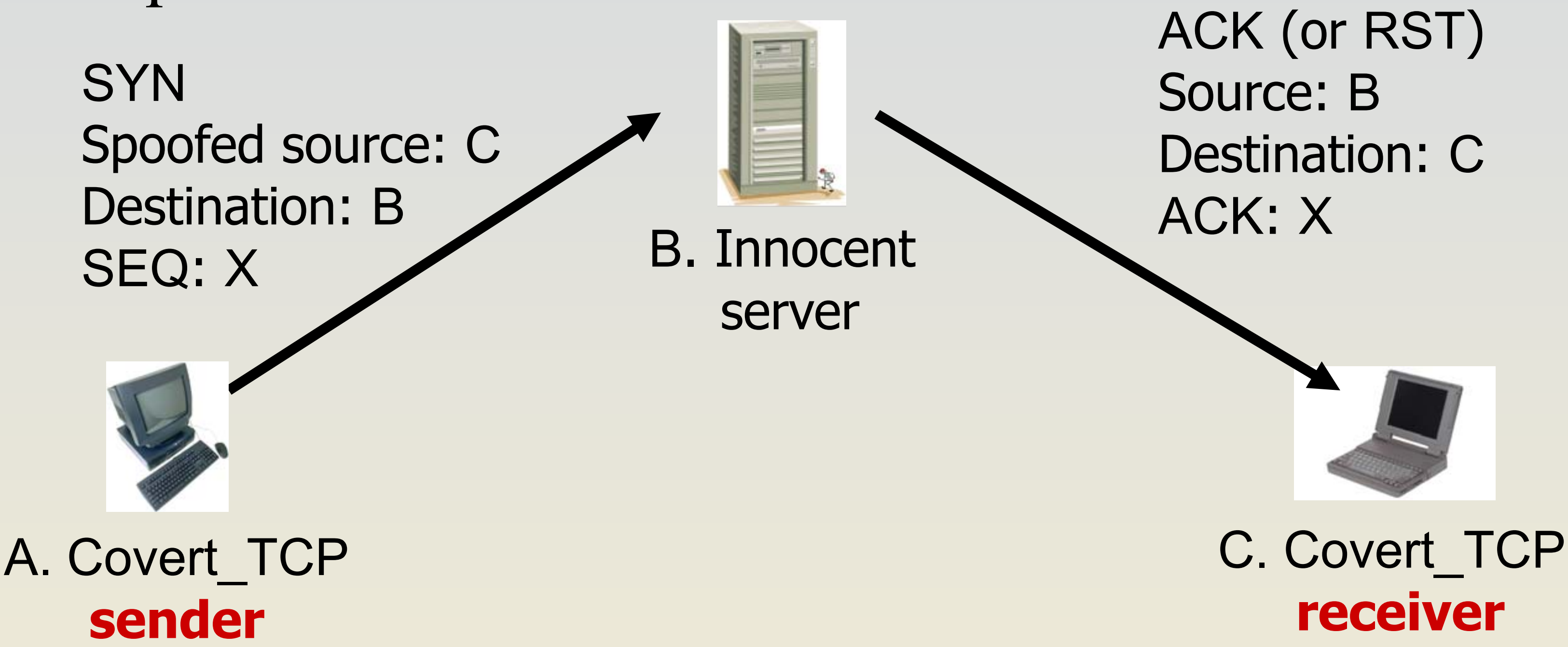
# Real-World Covert Channel



- Hide data in TCP header “reserved” field
- Or use covert_TCP, tool to hide data in
  - Sequence number
  - ACK number

# Real-World Covert Channel

- Hide data in TCP sequence numbers
- Tool: covert_TCP
- Sequence number X contains covert info



SYN
Spoofed source: C
Destination: B
SEQ: X

B. Innocent
server

ACK (or RST)
Source: B
Destination: C
ACK: X

A. Covert_TCP
**sender**

C. Covert_TCP
**receiver**