

Defeating Live Forensics in the Windows Kernel

Presented by Darren Bilby
AUSCERT 2006

Overview

- **Digital Forensics Acquisition**
- **The Live Imaging Process**
- **Windows Storage Architecture**
- **I/O Functionality in Rootkits**
- **DDefy**
- **DDefy Demo**
- **Better Methods for Live Imaging**



Security-Assessment.com – Who We Are

- **Specialist pure-play security firm**
- **Offices in Australia and New Zealand / Strong global partnerships**
- **Committed to research and improving our industry**
- **Specialisation in multiple security fields**
 - Security assessment
 - Security management
 - Forensics / incident response
 - Research and development



Digital Forensics Acquisition

- **Need to gather an evidential copy of a system**
- **The Aim**
 - Gather the “best” evidence available
- **Gather volatile information**
 - memory, process list, network connections, open files...
- **Power off machine and image disk**



Digital Forensics Acquisition

- **Two Competing Aims**
 - Gather the “best” evidence available
 - Allow the system to continue operation in an unhindered manner
- **Results in “Live Imaging”**
 - Taking a copy of a system while that system is still functioning in a live environment



Reasons for “Live Imaging”

- **Business critical systems that cannot be shut down**
- **Shutting down systems may create legal liability for examiners through:**
 - damaging equipment
 - unintentional data loss
 - hampering operations
- **Judge instructs that evidence gathering must be conducted using the least intrusive methods available**
- **Encrypted volumes**



Digital Forensics Acquisition

Live imaging is now “best practice” ...
...or at least common practice

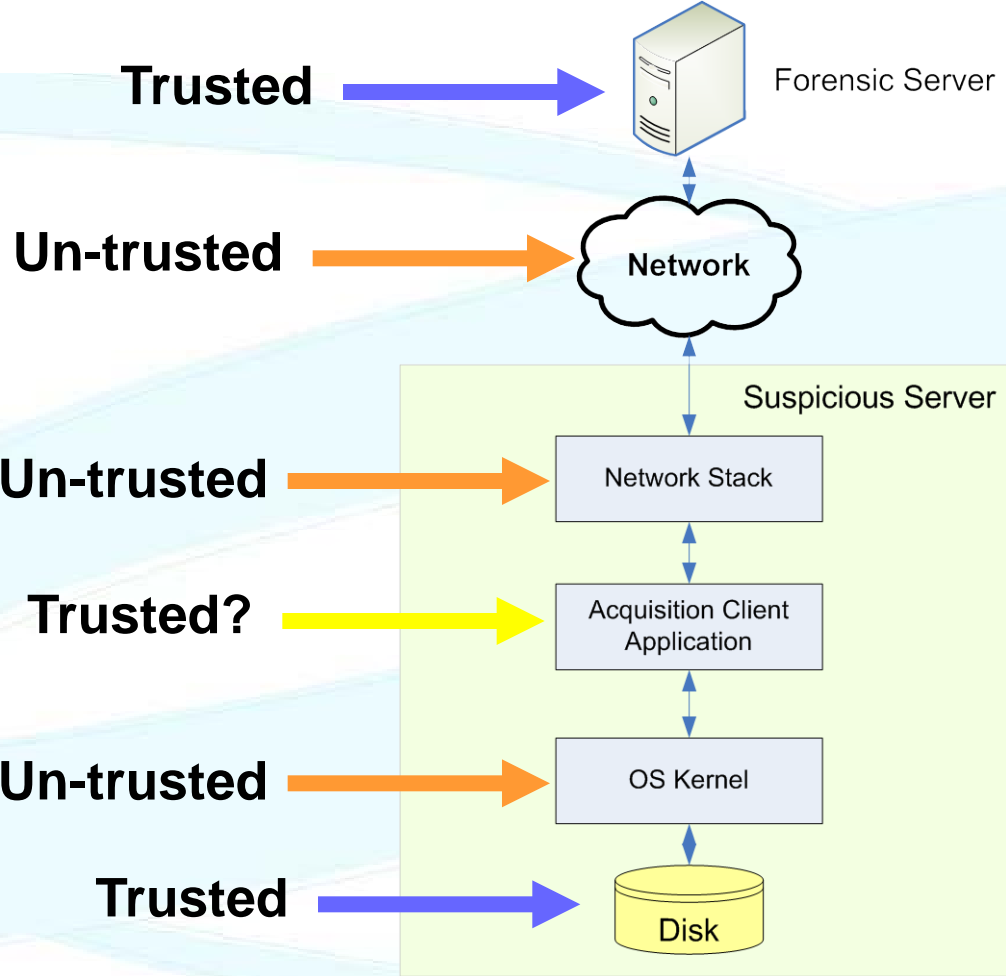
- **Tools**
 - Helix (dd/netcat)
 - Prodiscover IR
 - Encase EEE/FIM
 - FTK
 - Smart
 - ...



So this is common practice, accepted as legitimate by most courts of law, supported by many big name forensic vendors, it must be foolproof right?

uhhh... ok

The Live Imaging Process

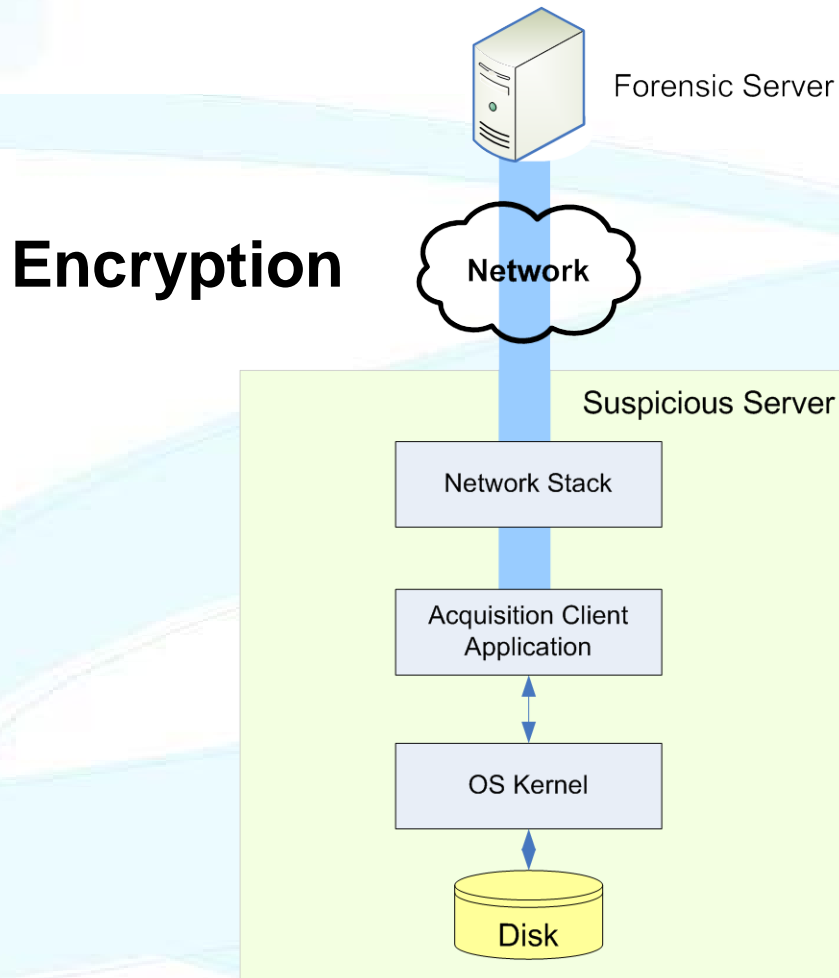


Live imaging...

... is like turning up to a homicide at the docks and asking the mafia to collect your evidence and take it back to the police station for you.



The Live Imaging Process



- **Encase**
 - SAFE public key encryption architecture
- **DD**
 - Cryptcat
- **Prodiscover IR**
 - Twofish Encryption



Live imaging...

... with network encryption is like turning up to a homicide at the docks and then asking the mafia to collect your evidence. Then handing it to an elite military squad to take it back to the police station for you.



Live Imaging

- **How do we know we have collected all the original evidence on an un-trusted system?**



Live Imaging on Windows Overview

- What happens when you read a file?
- Rootkit functionality for disk IO
- What happens when you run dd or FTK imager



Windows Storage Architecture

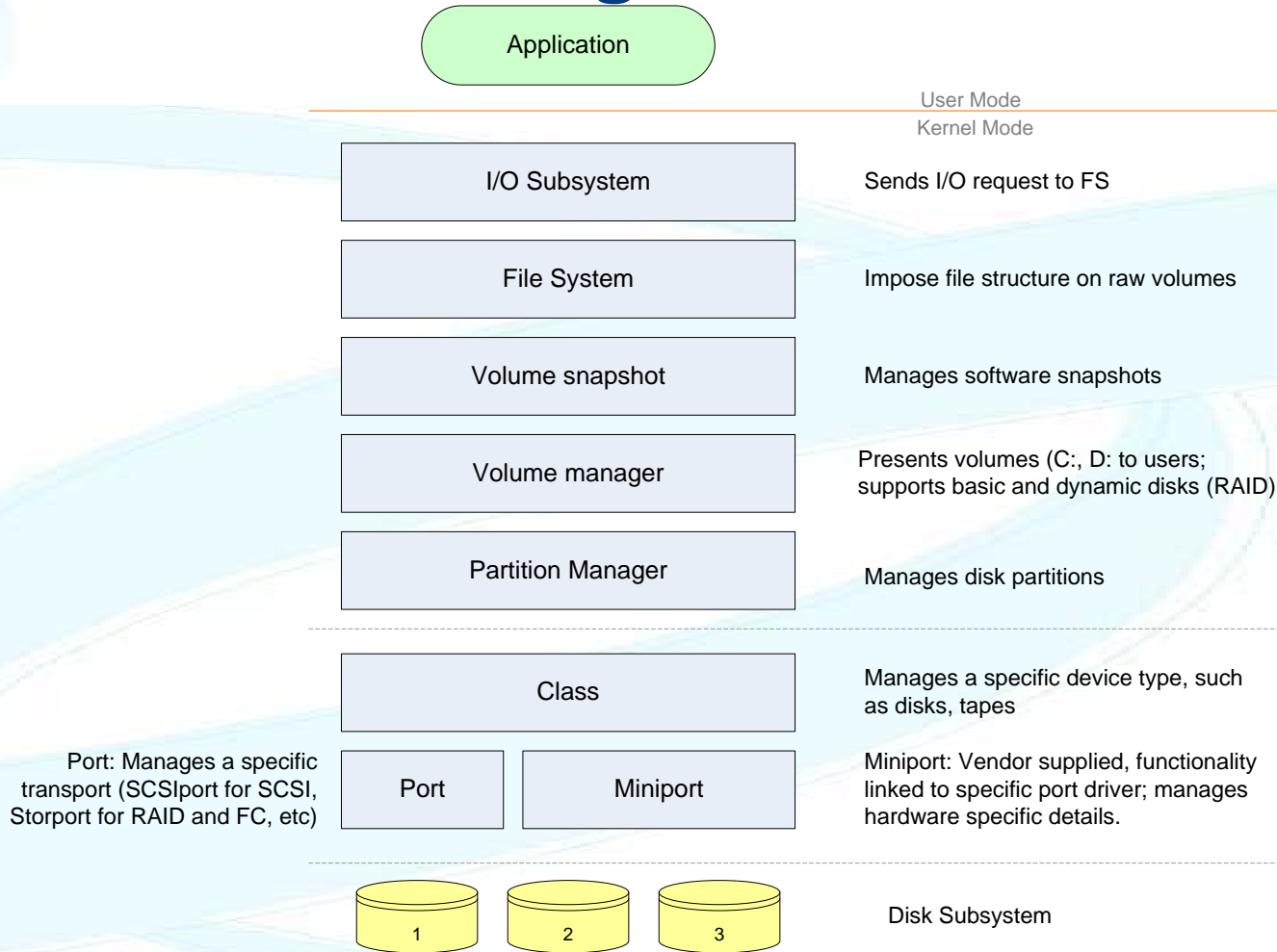
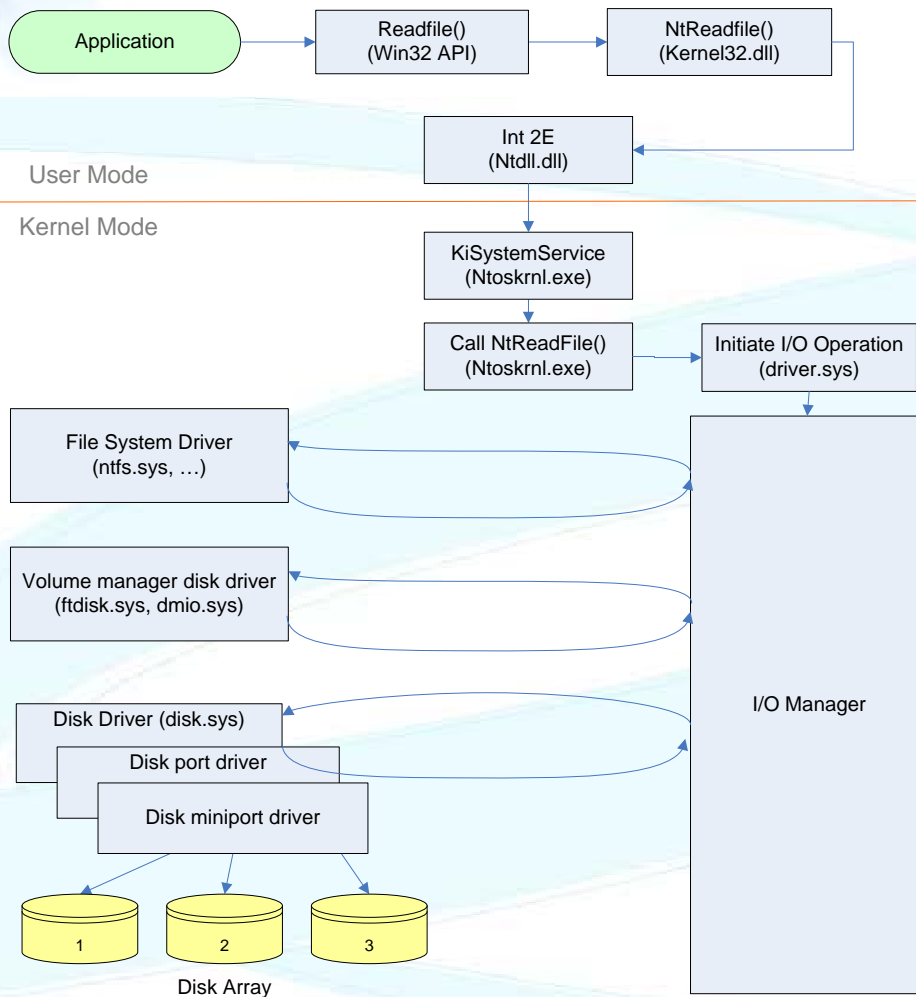


Diagram adapted from Microsoft
Windows Internals Fourth Edition



What Happens When You Read a File?



- Readfile() called on File1.txt offset 0
- Transition to Ring 0
- NtReadFile() processed
- I/O Subsystem called
- IRP generated
- Data at File1.txt offset 0 requested from ntfs.sys
 - translation
- Data at D: offset 2138231 requested from dmio.sys
 - translation
- Data at disk 2 offset 139488571 requested from disk.sys



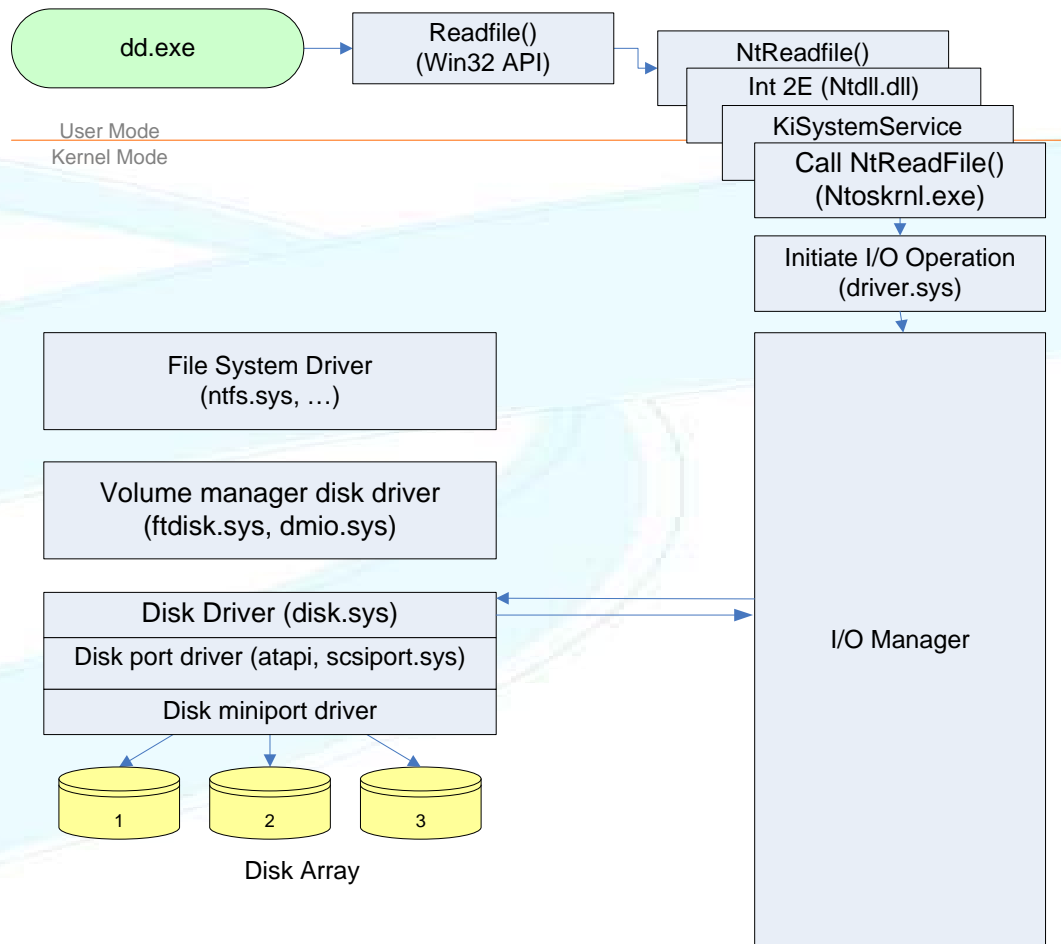
Live Imaging with DD

- Live Imaging Command
dd.exe if=\\.\PhysicalDrive0
of=z:\physicaldrive0.raw.dd

- \\.\PhysicalDrive0 is a device symbolic link to the raw disk

- The File System Driver and Volume Manager Driver are bypassed

- This method has been confirmed as used by
 - DD (GM Garner)
 - FTK Imager
 - Prodiscover IR



Rootkits

Rootkits

- **Malicious people want to remain undetected on a system**
 - Operating system must be subverted to give a false view
 - Hide files, processes, network traffic

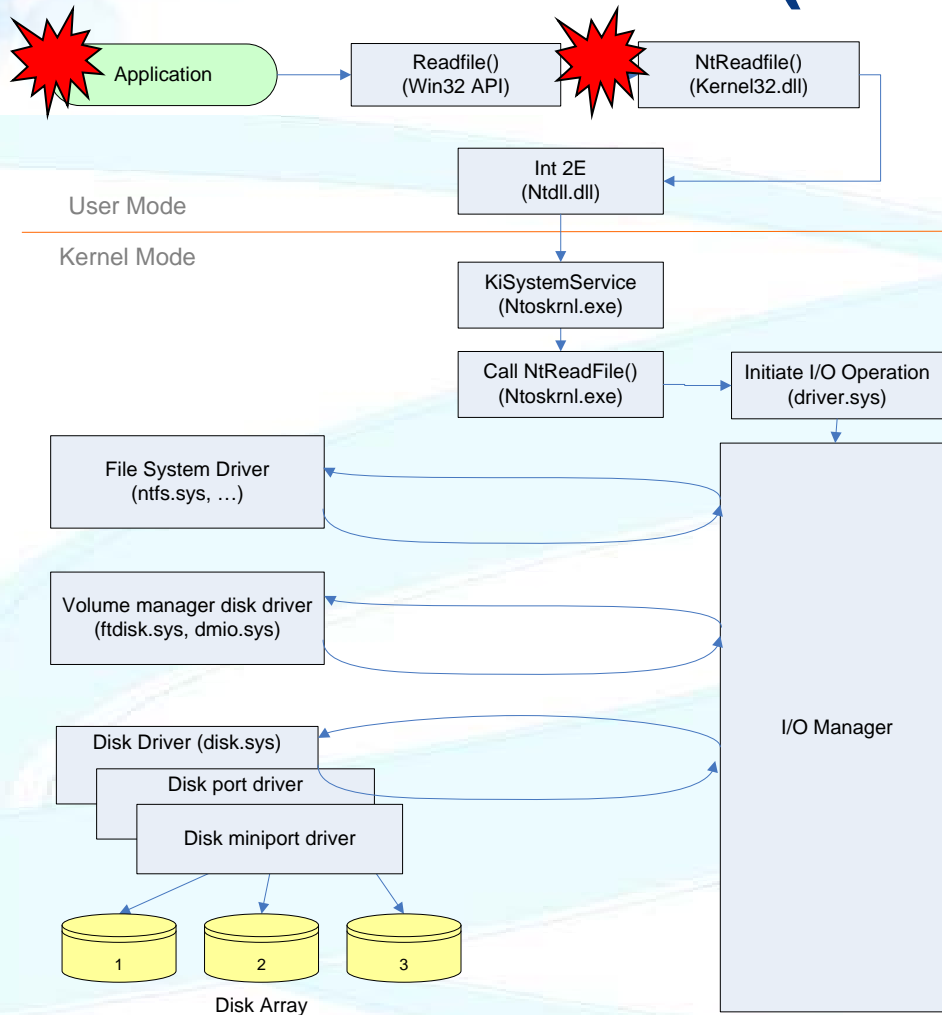


Rootkits

- **Dangerous for incident responders and security people because:**
 - No Discovery - If an incident is not discovered it will never be investigated.
 - The Trojan Defence – If it cannot be proved that a rootkit was not present, a case may be undermined.
 - Evidence tampering – An investigator cannot trust any information gathered from the machine.

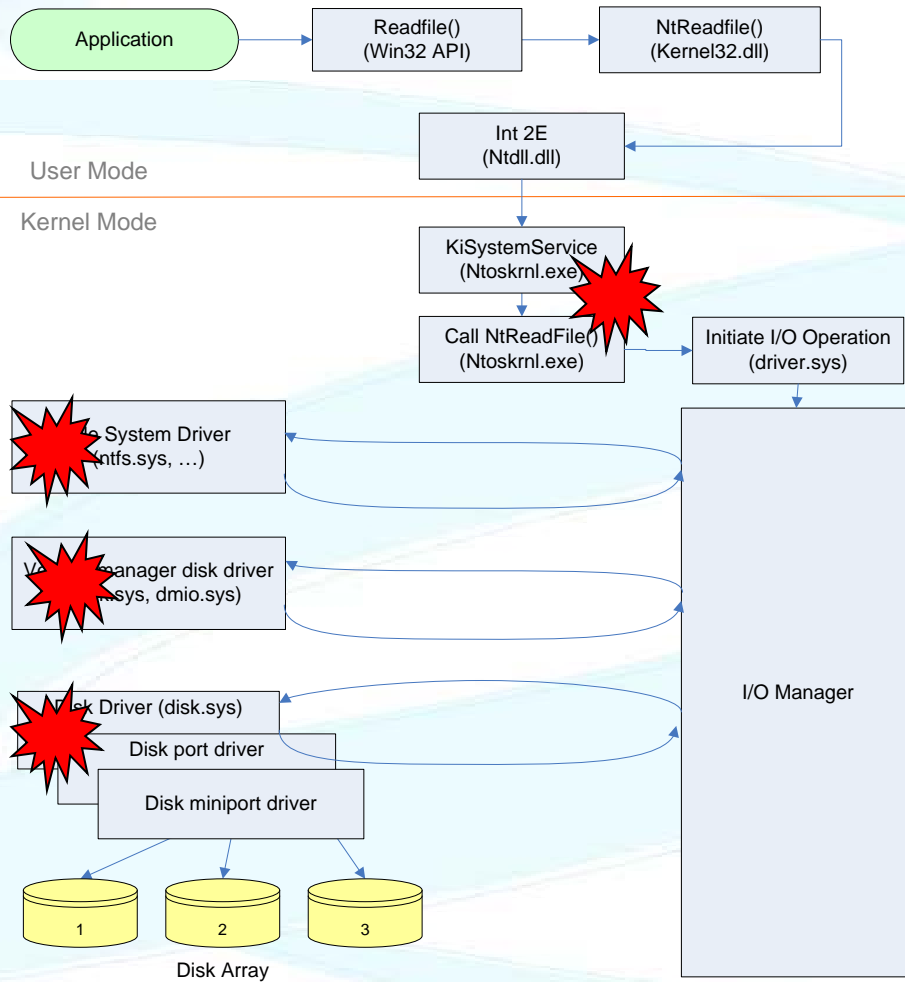


Public Userland (Ring 3) Rootkits



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

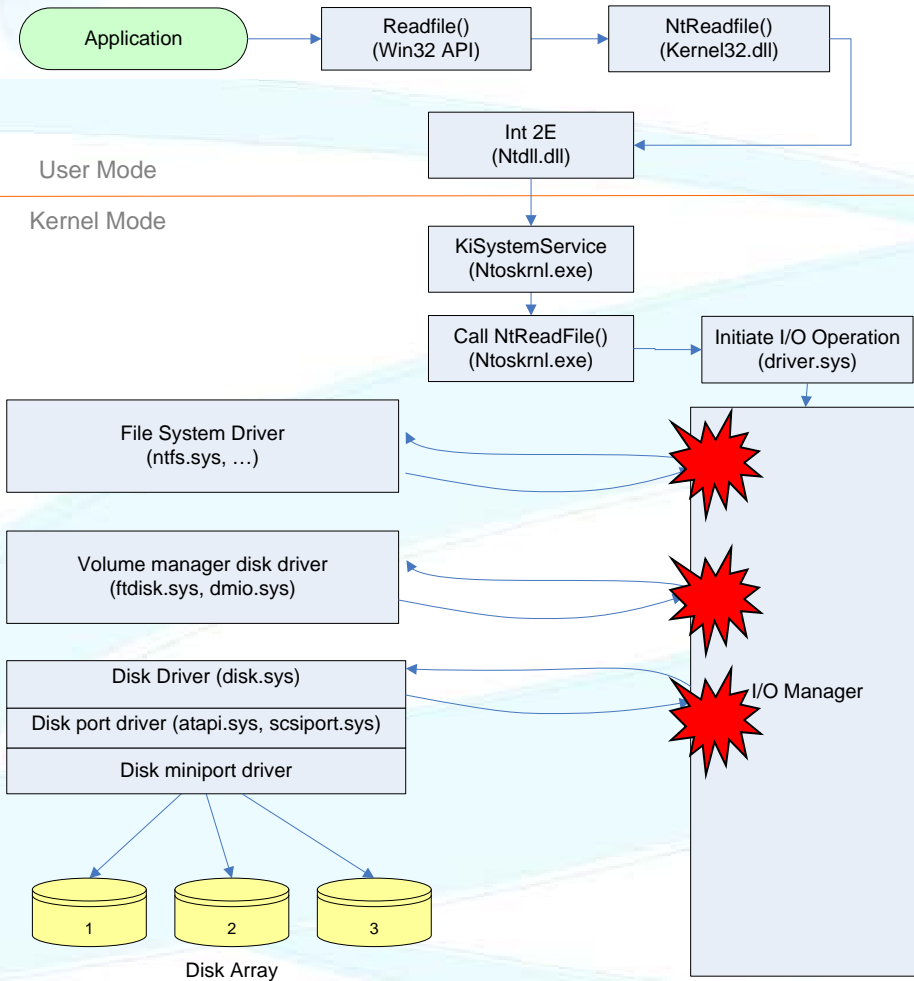
Kernel (Ring 0) Rootkits



- **Kernel Hooking**
E.g. NtRootkit
- **Driver replacement**
E.g. replace ntfs.sys with ntfs.sys
- **Direct Kernel Object Manipulation – DKOM**
E.g. Fu, FuTo



Kernel (Ring 0) Rootkits

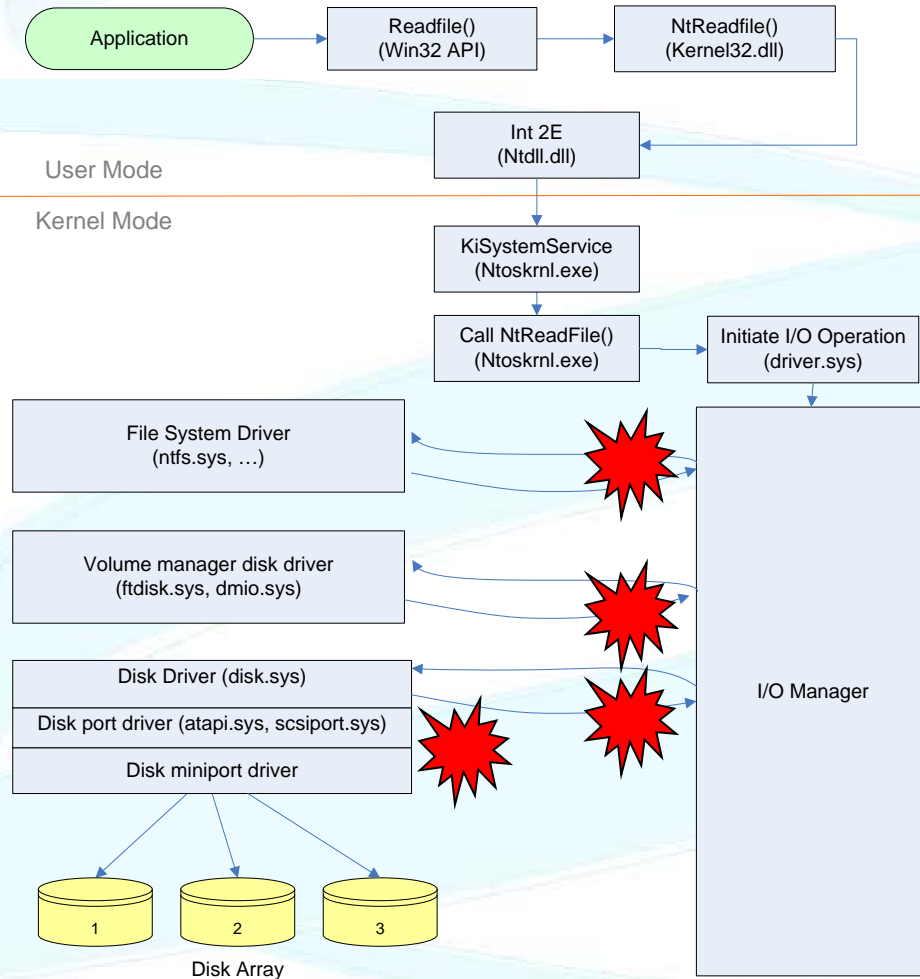


- **I/O Request Packet (IRP) Hooking**
 - IRP Dispatch Table

E.g. He4Hook (some versions)



Kernel (Ring 0) Rootkits



- **Filter Drivers**
 - The official Microsoft method
- **Types**
 - File system filter
 - Volume filter
 - Disk Filter
 - Bus Filter

E.g. Clandestine File System Driver (CFSD)



That's great... but why is this interesting?

It's interesting because...

- If we can identify bits on disk as relating to a file we have opportunity
- There are many places to subvert the file read process
- It is very unlikely to be detected
- This gets an attacker closer to the trump card for the “whoever hooks lowest wins” arms race

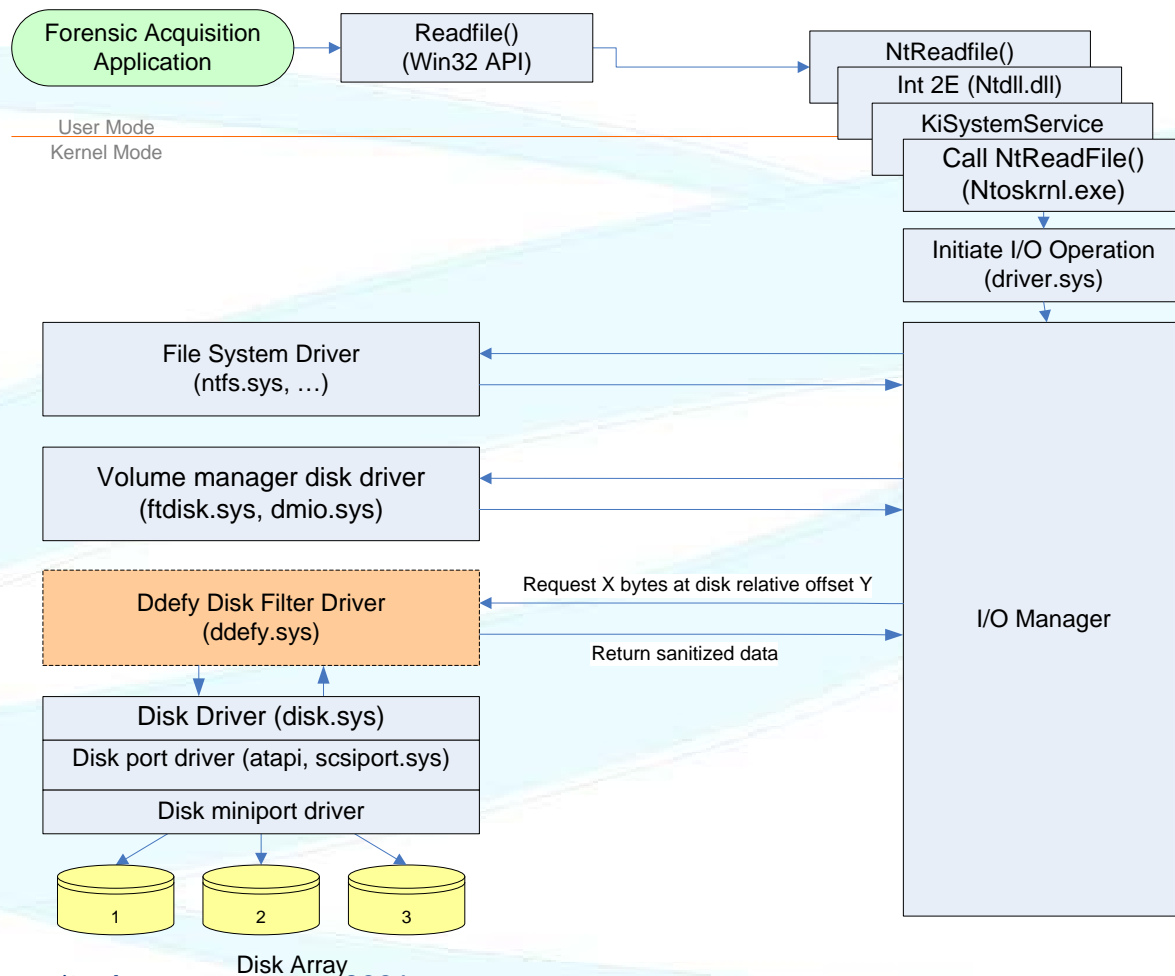


DDefy

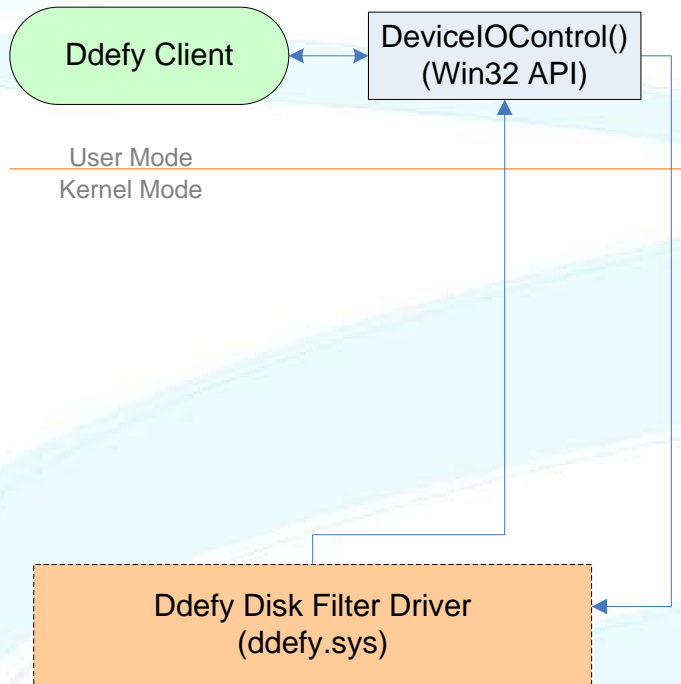
- **The Aim: When someone forensically analyses my machine, they should get a valid image, but not my sensitive data.**
- **Written on the power of short blacks and jack daniels**
- **Proof of concept for 2K/XP/2k3**
- **Standard Upper Disk Filter Driver**
- **Intercepts IRP_MJ_READ I/O Request Packets sent to the disk and modifies the return data**
- **No hooking, DKOM or other modification**
- **Hiding in plain sight**
- **Can be found in device manager**



DDefy: Where It Lives



DDefy: The Process



1. Determine drive info and NTFS characteristics for partition
2. Determine Filename and Directory entry position on disk
3. Determine clusters containing file data and their position on disk
4. HideData(Disk 0, Offset A, Length B, replace with Nulls)
5. HideData(Disk 0, Offset C, Length D, replace with "fakefile.txt")



Demo

[D:\video\ddefypres\ddefy-noddefy
imaging.wmv](#)

[D:\video\ddefypres\ddefy-
analysiswithddefyinstalled.wmv](#)

DDefy Results

- **Any data that is stored on the physical disk can be hidden from the live forensics tool**
- **There is no way to completely prevent this**
- **Live forensic imaging is still a useful tool**
 - but it needs to be used with full knowledge of the implications
- **Image the disk offline whenever possible**
- **Memory analysis becomes very important**

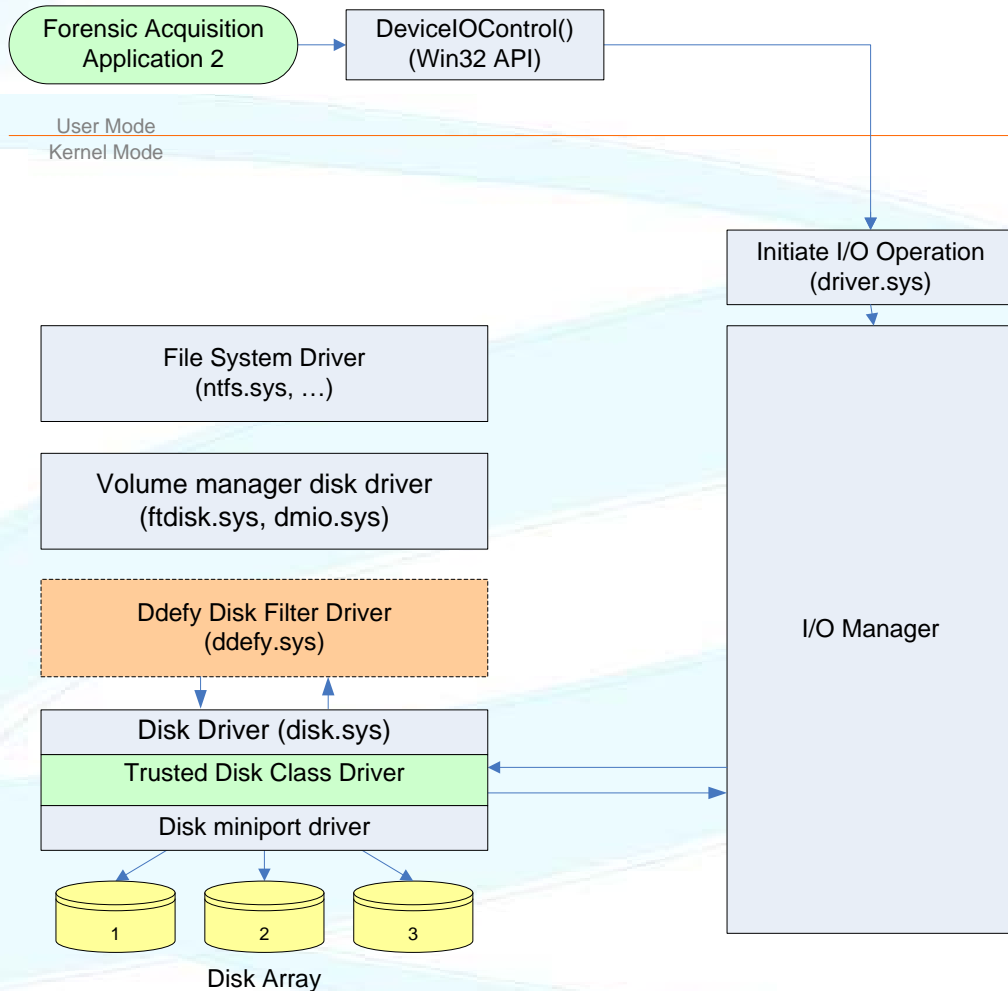


DDefy Challenges

- **Replacing Data without Corruption**
 - MFT replacement
 - Null, Random, Bad Sector, Deleted, Random, Other Files
- **Windows Disk Caching**
 - If the file system has cached a file, the disk won't be asked for the data



A Better Way of Acquiring Data



- **Method**

- Install trusted disk class driver
- Communicate directly with driver using DeviceIOControl
- Encrypt communications between driver and application

- **Challenges**

- Stability
- OS Specific



Live imaging...

... using a trusted driver and direct I/O is like turning up to a homicide at the docks and collecting the evidence yourself, while the mafia stand over you.



Further Research

- **Effects on rootkit detection tools**
- **Applying the same techniques to memory forensics**
 - Intercepting dd if=\\.\PhysicalMemory
 - Shadow Walker – (S. Sparks, J. Butler)
- **Implementation of an open source direct IO driver**



Questions ?

<http://www.security-assessment.com>

darren.bilby@security-assessment.com



Resources

- **Windows System Internals 4th Edition**– D. Solomon, M. Russinovich
- **Rootkits** – G. Hoglund, J. Butler
- **Primary Windows Rootkit Resource**
<http://www.rootkit.com>
- **Joanna Rutkowska – Stealth Malware Detection**
<http://www.invisiblethings.org>
- **Windows Driver Development Resource**
<http://www.osronline.com>

