**Forensic tools**

Inhoud---------------------

[Part 1: The difficulty with digital forensics 6](#_Toc375486317)

[Volatility (beweeglijkheid) 6](#_Toc375486318)

[Forensics is about data recovery 7](#_Toc375486319)

[Data recovery 7](#_Toc375486320)

[Anti-forensics 7](#_Toc375486321)

[Tools or not ? 7](#_Toc375486322)

[Part 2: Hardware security 7](#_Toc375486323)

[Cameras 7](#_Toc375486324)

[CCTV (Closed Circuit TV) 8](#_Toc375486325)

[Physical / hardware IT Security 8](#_Toc375486326)

[Trusted Platform Module (TPM) 8](#_Toc375486327)

[The big downside of TPM/TXT 8](#_Toc375486328)

[Keyboard-Video-Mouse (Over IP) 8](#_Toc375486329)

[Local Console Access 9](#_Toc375486330)

[External media boot (hardening) 9](#_Toc375486331)

[Secure RAID 9](#_Toc375486332)

[RAID 1 10](#_Toc375486333)

[RAID 5 10](#_Toc375486334)

[RAID 5 Parity calculations 10](#_Toc375486335)

[RAID 10 10](#_Toc375486336)

[RAID 01 11](#_Toc375486337)

[Soft – or Hardware RAID? 11](#_Toc375486338)

[Examples for hardware RAID 11](#_Toc375486339)

[The big problem with HARDWARE RAID 11](#_Toc375486340)

[Software RAID 12](#_Toc375486341)

[The importance of Backup 12](#_Toc375486342)

[Backup systems 12](#_Toc375486343)

[SOLUTION: TAPE! 13](#_Toc375486344)

[Tape Libraries 13](#_Toc375486345)

[Backup policy! 13](#_Toc375486346)

[Backup Schemes 14](#_Toc375486347)

[Grandfather, Father, Son rotation 14](#_Toc375486348)

[Secure SAN configurations 14](#_Toc375486349)

[Native Storage Appliance. 15](#_Toc375486350)

[Storage Appliance Scheme 15](#_Toc375486351)

[A SAN without the hardware 16](#_Toc375486352)

[Making VSA HA 16](#_Toc375486353)

[Use case 17](#_Toc375486354)

[The VSA players 17](#_Toc375486355)

[Performance benchmark set-up 17](#_Toc375486356)

[Part 3: Hardware Security part 2 18](#_Toc375486357)

[Asset Management Software 18](#_Toc375486358)

[Hardware Data Recovery 18](#_Toc375486359)

[Unwanted hardware failures 18](#_Toc375486360)

[Part 4: Filesystems on Linux 19](#_Toc375486361)

[Logical structure of a hard drive 19](#_Toc375486362)

[MBR 19](#_Toc375486363)

[Cylinder Head Sector (CHS) 19](#_Toc375486364)

[Logical Block Addressing 19](#_Toc375486365)

[MBR Partition Entry 20](#_Toc375486366)

[EBR entry (512 bytes) 21](#_Toc375486367)

[Extended partition example 21](#_Toc375486368)

[Partition Layout Examples 22](#_Toc375486369)

[Part 5: Filesystems 22](#_Toc375486370)

[Blocks? 22](#_Toc375486371)

[Block groups 22](#_Toc375486372)

[A journaling filesystem? 23](#_Toc375486373)

[Filesystem reliability 24](#_Toc375486374)

[How does it work? 24](#_Toc375486375)

[Logging 24](#_Toc375486376)

[Ext3 Journaling modes 24](#_Toc375486377)

[Configuration 25](#_Toc375486378)

[Journal 25](#_Toc375486379)

[Ext4 25](#_Toc375486380)

[How does this help crash recovery? 26](#_Toc375486381)

[Forensic tools for data analysis (Linux) 27](#_Toc375486382)

[Sleuthkit 28](#_Toc375486383)

[File Allocation Table (FAT) 28](#_Toc375486384)

[FAT12, FAT16 & FAT32 28](#_Toc375486385)

[exFat (or FAT64) 28](#_Toc375486386)

[FAT Layout 29](#_Toc375486387)

[NTFS : MFT 30](#_Toc375486388)

[MFT records 30](#_Toc375486389)

[NTFS file attributes 31](#_Toc375486390)

[NTFS System Files 31](#_Toc375486391)

[NTFS Shortcuts? 31](#_Toc375486392)

[Symbolic vs hard vs soft links 31](#_Toc375486393)

[Multiple data streams 32](#_Toc375486394)

[Alternate Data Streams 32](#_Toc375486395)

[10 things to know about ADS 33](#_Toc375486396)

[Scenario 33](#_Toc375486397)

[Slack space (volume & file system) 33](#_Toc375486398)

[Information Concealment Engine? 34](#_Toc375486399)

[File System Insertion & Subversion Technique (FIST) 34](#_Toc375486400)

[File System Insertion Implementations 34](#_Toc375486401)

[NTFS compressed files 34](#_Toc375486402)

[NTFS Built-in Encrypted File System (EFS) 35](#_Toc375486403)

[NTFS Sparse Files (NTFS5 only) 36](#_Toc375486404)

[Partition Corruption 37](#_Toc375486405)

[MBR recovery? 37](#_Toc375486406)

[Partition boot sector recovery? 38](#_Toc375486407)

[Two kinds of boot loaders 38](#_Toc375486408)

[GPT 38](#_Toc375486409)

[GPT (U) EFI 39](#_Toc375486410)

[GPT Entry 39](#_Toc375486411)

[File Systems 39](#_Toc375486412)

[What is a file system? 39](#_Toc375486413)

[File Systems – LINUX 40](#_Toc375486414)

[Linux Extended File System 40](#_Toc375486415)

[Ext problems 40](#_Toc375486416)

[Ext Features 41](#_Toc375486417)

[Ext2, the most popular Unix File System. 41](#_Toc375486418)

[Ext2 permissions 42](#_Toc375486419)

[Deleting a file on ext. 42](#_Toc375486420)

[Part 7: Windows Analysis using the Registry 42](#_Toc375486421)

[Registry: A wealth of information 42](#_Toc375486422)

[Registry History 43](#_Toc375486423)

[Definition 43](#_Toc375486424)

[Registry Organization 43](#_Toc375486425)

[Windows 9x Registry Files 44](#_Toc375486426)

[Windows XP Registry Files 45](#_Toc375486427)

[Windows 7/8 Registry files 45](#_Toc375486428)

[Windows Security and Relative ID 46](#_Toc375486429)

[Most Recently Used (MRU) 47](#_Toc375486430)

[Registry Analysis 47](#_Toc375486431)

[Case study: lezen dia 38 48](#_Toc375486432)

[Registry Forensics Investigation 48](#_Toc375486433)

[User Login 49](#_Toc375486434)

[Create a log! 50](#_Toc375486435)

[Part 8: Command Line tools 50](#_Toc375486436)

[Command line 50](#_Toc375486437)

[Net – command 50](#_Toc375486438)

[NETSTAT 51](#_Toc375486439)

[NETSH 51](#_Toc375486440)

[NETSH examples 52](#_Toc375486441)

[Windows command prompt 52](#_Toc375486442)

[Advantages / disadvantages 52](#_Toc375486443)

[PowerShell Basics (°2006) 53](#_Toc375486444)

[Aliases 53](#_Toc375486445)

[Scripting in PowerShell? 53](#_Toc375486446)

[FORMS 54](#_Toc375486447)

[Examples 54](#_Toc375486448)

[PowerShell Tools 54](#_Toc375486449)

[Other tools 55](#_Toc375486450)

[Linux Bash 55](#_Toc375486451)

[Linux Script 55](#_Toc375486452)

[Part 9: Domain Name Service (DNS) 56](#_Toc375486453)

[Terminology 56](#_Toc375486454)

[How does it work? 57](#_Toc375486455)

[How does a DNS packet look like? 57](#_Toc375486456)

[Issues with DNS 58](#_Toc375486457)

[Attack Vectors 58](#_Toc375486458)

[The Time-To-Live (TTL) 58](#_Toc375486459)

[Attack example 58](#_Toc375486460)

[Kaminksy attack 59](#_Toc375486461)

[However? 59](#_Toc375486462)

[Better solution? 59](#_Toc375486463)

[How does DNSSEC work 60](#_Toc375486464)

[Forensic use 60](#_Toc375486465)

[DNS cache 60](#_Toc375486466)

[DNS cache clear? 61](#_Toc375486467)

# Part 1: The difficulty with digital forensics

A certified forensics expert must be a member for a law enforcement agency.

Evidence 🡪 is critical

Aim?

* Identify
* Preserve
* Recover
* Analyze
* Present

Techniques?

* Media analysis
  + Offline analysis and scanning of possibly incriminating data on storage media
* Live analysis
  + Extracting evidence while on a running system from within the operating system
* Deleted files
  + Recovery of deleted data.
* Steganography
  + Is a technique to hide data in other data. This can be within files (ex. Pictures) or in metadata (headers) or even within file systems.

## Volatility (beweeglijkheid)

The worst enemy of a forensic analyst!

The most important step = seizing evidence

(example: data only exists in RAM, after shutdown it’s lost!)

Forensic tools!

* Commercial and open-source tools
  + Open source may not be used in certain countries
  + Can be used in Belgium if WELL DOCUMENTED
* When is software forensically correct?
  + No altering of the data: read-only is read-only
* Solution? Forensic imaging software
  + For example: encase Forensic V7
  + Or open source
* Making sure that the data is not touched:
  + Need for forensic write blockers

## Forensics is about data recovery

Lots of storage media

* Hard drives / SDD’s
* Compact disk, DVD, floppy
* Tape, backups
* Ram memory
* SD-cards
* USB drives
* Smartphones & tablets
* Also CLOUD STORAGE

## Data recovery

* Tricky and hardware dependent
* Should not (or as little as possible) be done on a live system!
* Understanding of underlying structures is critical
* Data is very persistent

## Anti-forensics

* Secure data handling
* More than just data removal
* If a forensic analyst can find data you want hidden, anyone can!
* Forensic analyst are limited in time!
  + If the recovery takes longer: data stays hidden!

## Tools or not ?

Lots’s of tools

* Mainly open source
* Both linux & Windows Tools

But more important: hand work

* Know what you are doing is important!
* Never execute a command if you do not know what it will do

# Part 2: Hardware security

## Cameras

* Preferably connected to a separate network. If the footage is to be stored, make sure this is also secure. Use a rotation system!
* Every individual that is recorded has the explicit right to watch his footage
* Security cameras must be announced by a special, official pictogram

## CCTV (Closed Circuit TV)

* It’s illegal to record images from people who aren’t aware!

Locations for which the rules differ:

* Non clused location: any location that is free to the public
  + Only under supervision by law enforcement
* Closed location that is accessible for the public (warehouse, museums)
  + May not store for longer than a month
  + All cameras must be declared to the official instances
* Close location that is not accessible for the public

## Physical / hardware IT Security

* Kensington lock
* TPM / TXT / Local
* External media startup
* Backup
* RAID
* Virtual Storage Appliances (HA)

## Trusted Platform Module (TPM)

* Standard organized by Trusted Computing Group
* ISO / IEC 11889
* Among others, it can offers a secure RSA key generator
  + Bitlocker can use this!
* It creates a hash-key summary of the hardware configuration

## The big downside of TPM/TXT

Security check occurs at boot time! 🡪 A running system is still vulnerable

**Local Console Access (important!)**

* KVM
* Local logon, disable Windows Local logon
* Lockdown mode (ESXi)

## Keyboard-Video-Mouse (Over IP)

* Easy local console access to all your servers
* Also over IP (web interface with ActiveX/Java console)

## Local Console Access

Windows : disable local access for users other than admins on all server members.

* Allow login for admins (or one admin) for troubleshooting purposes

Linux: same, but must be configured locally;

* Disable all tty consoles except one!
* Edit the file /etc/security/access.conf
* Add line:
  + -:ALL EXCEPT root:LOCAL (to disallow local login for all users except root)

Local Linux configurations can be executed more securely and better using Configuration Management tools like Puppet (Linux) or inTune (MS)

## External media boot (hardening)

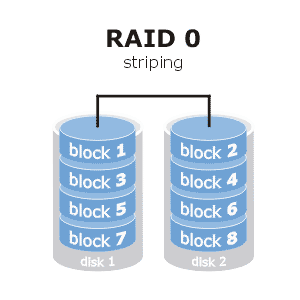
Configure the BIOS of your system

* Add a BIOS password
* Configure the boot order; only boot from local hard drive, do not try other media

After that, put a padlock on your system to prevent CMOS reset.

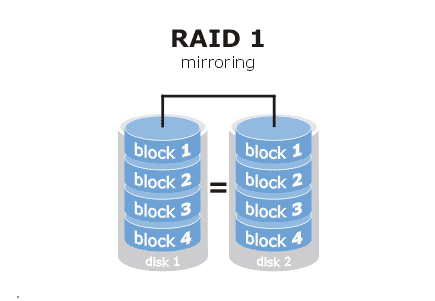
## Secure RAID

RAID = Redundant Array Of Independent (or Inexpensive) Disks.

**RAID 0**: stripping, secure against theft, unsecure against hardware failure.

## RAID 1

= mirroring (using 2 or more disks)

* Safe against hardware failure, completely unsafe against theft
* When using RAID 1: use volume encryption (e.g. bitlocker)

Hot spare disk = idle disk that has spun down, if needed a rebuild action can start automatically

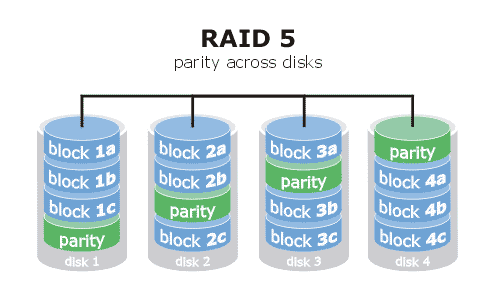
Introduction of Redundancy factor:

* For 2 DISK RAID 1: 2 /2 + 0/ 1 = 1
* For 3 DISK RAID 1: 3/3 + 2/2 + 0/1 = 2

## RAID 5

Parity, at least three disks needed, always one disk lost for capacity

* If one disk fails, we are left with a RAID 0 alike operation
* If Hot Spare is configured, rebuild action starts immediately (bigger load on remaining disks!)

****

Redundancy factor:

* 3 Disks in RAID 5: 3/3 + 0 = 1
* 4 Disks in RAID 5: 4/4 + 0 = 1
* 4 Disks in RAID 5 + spare = 4/4 + 4/4 + 0 = 2

## RAID 5 Parity calculations

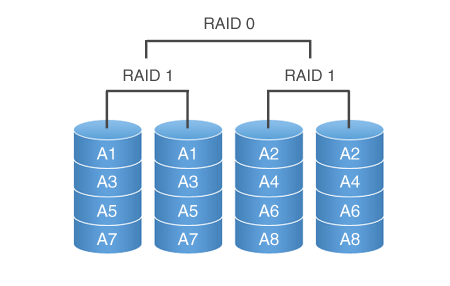
Parity mechanism is based on XOR calculations

Current (fast) processors make RAID 5 as fast as RAID 1 or better.

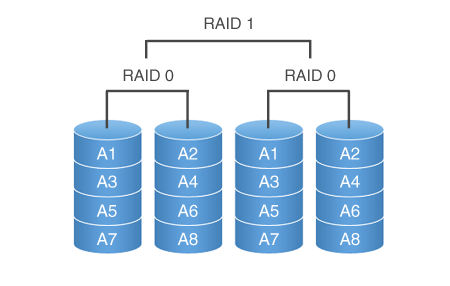
Bit example:

* File to write: 0010 1010 1000 1110 to RAID 5 on three disks
  + Disk 1: 0010 1010
  + Disk 2: 1000 1110
  + Disk 3: -------------- (parity)
* Parity Calculation: 0010 1010 XOR 1000 1110 = 1010 0100
* Etc.

## RAID 10

* At least 4 disks are needed, capacity is cut in half.
* Most Frequently used by enterprises
* Read like this: RAID 1 + RAID 0
* Redundancy factor:
  + 4 disks = 4/4 + 2/3 + 0/2 = 1.66…
  + 6 disks ( 2 disk RAID 1): 6/6 + 4/5 +2/4 + 0/3 = 2.3
  + Exercise (see ppt)

## RAID 01

****What is wrong with RAID 0 + RAID 1

* Same capacity as with RAID 10!
* Different redundancy factor

Let’s calculate the redundancy factor:

* 4 disks: 4/4 + 1/3 = 1,33… (as opposed to the 1.666 with RAID 10)

## Soft – or Hardware RAID?

Difference: where are the RAID calculations taking place ?

* Hardware RAID: dedicated RAID chip (onboard or plugin card)
* Software RAID: shared processor (usually CPU)

If working with DAS systems: hardware RAID has not much use!

* MAIN CPU is more than strong enough (if not stronger) for added RAID Functionality
* Recovery is easier with Software RAID (forensic analysis)

With SAN/NAS there usually is not much choice: dedicated RAID controller with proprietary algorithms.

## Examples for hardware RAID

EMC RAIDS (or parity RAID): == RAID 5 with a dedicated parity disk (on volume level)

RAID 1.5 == mirroring and striping at the same time

* RAID 1 for write, RAID 0 for read
* Linux and Solaris software RAID do this by default with RAID 1

RAID 5E, 6E, etc… = RAID 5 or 6 with integrated hot spare (E = enhanced)

* Not using a dedicated hot spare, but spare blocks are distributed. In 4 disk RAID 5E, every disk contains 50% data, 25% parity and 25% spare blocks.
* Failure results in ‘regular’ RAID 5, however rebuilt is completely done on “old” disks and is very I/O intensive

RAID Z, RAID K, X-RAID , UnRAID etc.

## The big problem with HARDWARE RAID

* Besides the algorithms and protocols being proprietary and therefor very difficult to recover in case of data loss or forensic research.
* Every (good) RAID controller stored RAID configuration on the disks for recovery purposes…

However: there’s the aging of RAID controllers

* For example: RAID 10 on Areca Raid controller is not recognized if the disks are connected to a Adaptec RAID controller!
* OR: RAID 10 on Areca RAID bought 5 years ago, is not recognized in recent ARECA RAID.
* Hence the single point of failure with hardware RAID : RAID controller!

The SSD problem (later)

## Software RAID

Can be done by the OS itself

* Windows Dynamic Volumes or Storage Spaced
* Linux soft raid or Logical Volumes

Or by extra software (StarWind)

* Highly flexible (modern systems can even use thin provisioning)
* Usually easy expandable
* Recoverable (usually backwards compatible)
* “free” (included in the OS)
* Works independently of interface (IDE, SATA, SAS, SCSI, …)
* Fast (use modern, powerful CPUs)

However, OS not easy to install onto software RAID

Not all RAID levels are possible

Hot Swap not always supported (depends interface protocol configuration)

## The importance of Backup

* RAID is NOT backup

Top reasons for data loss:

* Hardware / system failures
* Human error
* Software corruption
* Viruses
* Natural disasters

Backups are also handy when errors are detected that have been introduced in older systems (and not immediately detected)

## Backup systems

HDDs are cheap, have big capacity and highly portable

* Can be encrypted
* However: corporate data (HR contracts, etc…) must be available for a period of 15 years 🡪 HDDs can fail to live that long
* HDDs are very fragile, one drop of 10cm is almost certain death for a spinning disk
* Not ideal for long time storage

Optical media (CD/DVD/BluRay) is also cheap and highly portable

* However: there file expectancy depends on the fabrication (sometimes only a year)
* Plus they do not have a high capacity

## SOLUTION: TAPE!

LTO ( Linear Tape-Open)

= magnetic tape storage with open standard (important if it must be readable within 15 years).

Supported by ie. HP, IBM and Seagate

Many different generations of LTO (LTO-2 is mostly used)

* LTO-5 is most recent (2010) and offers 1,5 TB / tape (uncompressed).
* LTO – 6 was released in December 2012 and offers 2,5 TB / tape (uncompressed)

Can be configured as WORM (Write-Once-Read-Many) for archiving reasons

Life expectancy of LTO-2 is 21 years when written once a month.

LTO protocol ensures automatic and immediate verification of the written data.

## Tape Libraries

Tapes usually have physical write security (read-only switch)

Tape libraries are devices to be placed in racks, they store backup tapes and rotate active and passive tapes. Allowing safe and secure removal.

Connected to a backup-server via DAS (external SAS or SCSI).

Cheaper (manual) alternative: HP DAT tapes

## Backup policy!

When will you create backups? Every 10 minutes? Every day? Every week?

How about your coworkers delete a DB that has been used up to the last minute?  
How about a virus that has been roaming for two weeks / months?

Where will you store them? All on premises? When and wich backup will you take home, what do you do with it ?

## Backup Schemes

FIFO is simplest

* For example: 21 daily backups, oldest one gets discarded every time.

Used to keep the longest possible tail of daily backups

Used when archived backups are not as important (no need to go back one year).

Useful when data before rotation period is irrelevant

Big downside: if an error is detected that has been around for 2 months, all backups are useless.

## Grandfather, Father, Son rotation

You have a box labeled “daily” with 14 tapes.

* You create a full backup daily

You have a box labeled “weekly” with 12 tapes.

* Every week you take the last daily tape and put it in the box “weekly”, this is the father, removing the oldest weekly
* Placing a new tape (tape renewal) into the box “Daily”

You have a box labeled “Monthly” with 12 tapes.

* Every month you take the last weekly and put it in the box “monthly”, this becomes the grandfather.

Etc..

Every week , take the latest tape home and put it in a vault. DO NOT STORE THEM IN YOUR CARE / BACKPACK.

## Secure SAN configurations

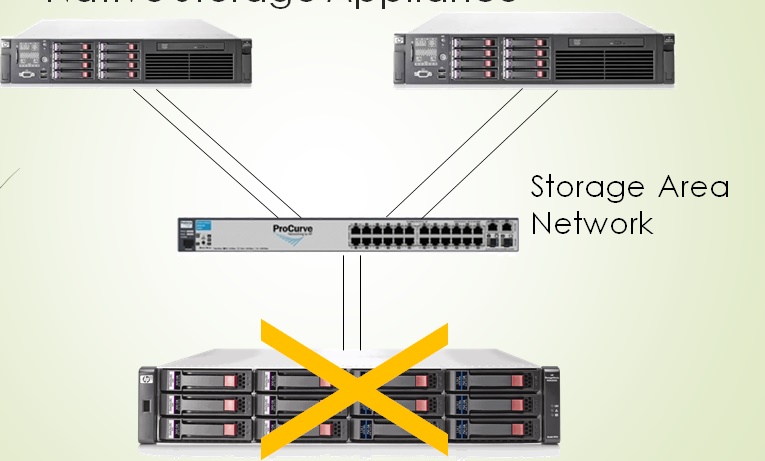
Creating a powerful, reliable and secure SAN environment does not have to be costly!

Solutions for KMO’s: (Virtual) Storage Appliance (VSA)

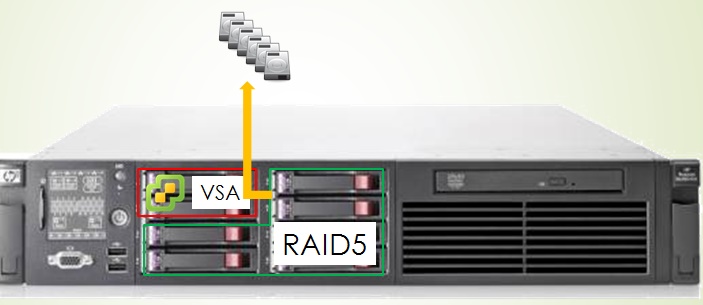
Convert any (old?) hardware to a SAN.

Examples: Open- E, Nexenta, OpenFiler, Starwind, …

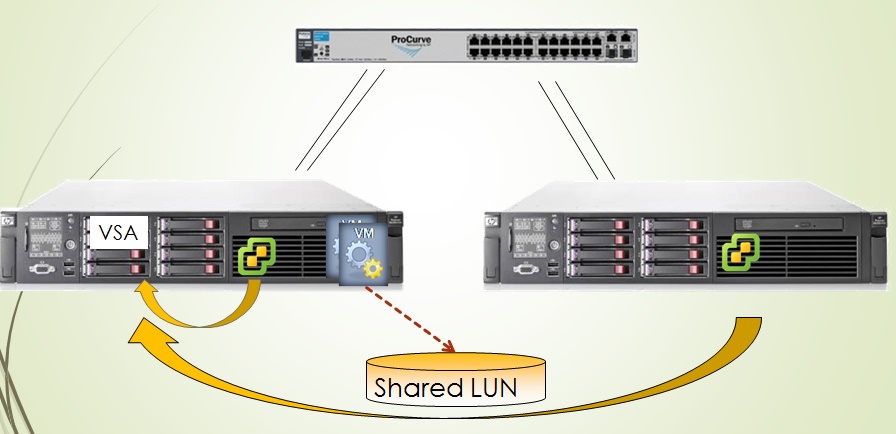
## Native Storage Appliance.



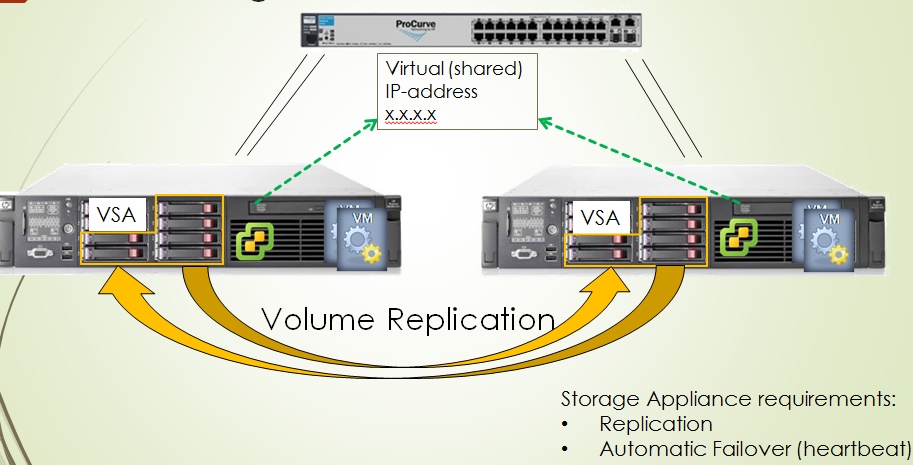
## Storage Appliance Scheme



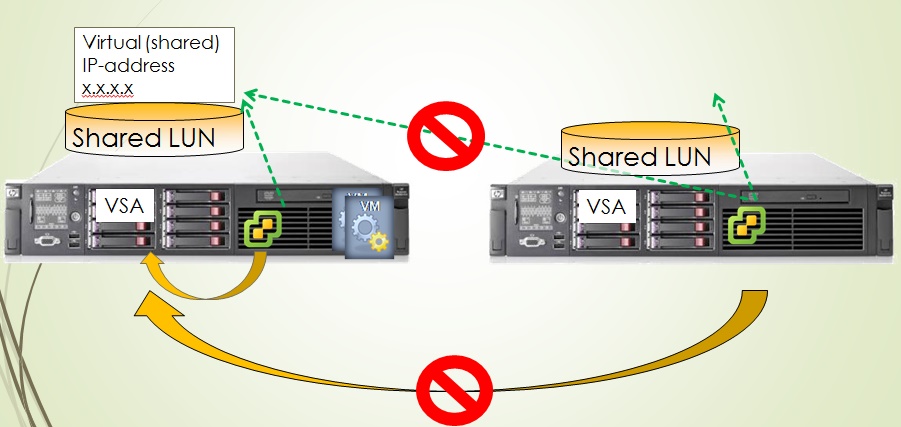
## A SAN without the hardware



## Making VSA HA



## Use case



## The VSA players

* HP StorageWorks
* Datacore SanMelody
* VMware VSA
* Gluster (only NAS)
* ….

How good is the performance ? Let’s find out.

## Performance benchmark set-up

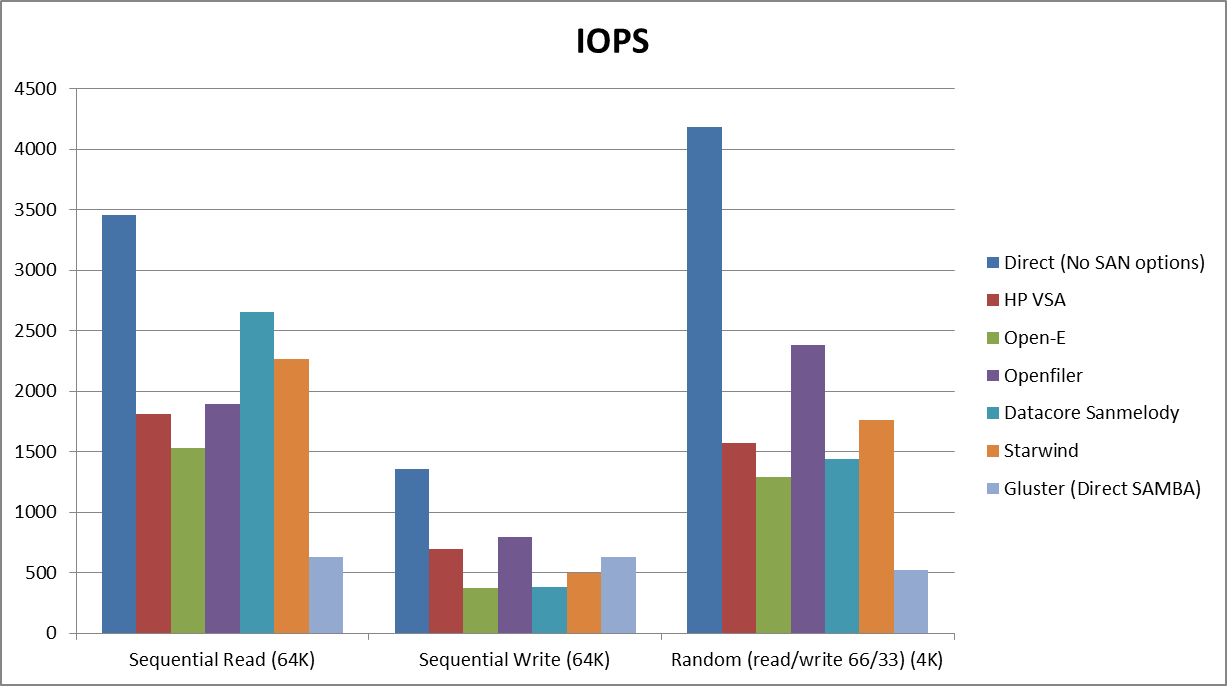
SAN Appliance = 2 vCPU & 2GB with 2 iSCSI LUNs

VM on big SAN LUN = Windows 2003 Enterprise x86 running dynamo.exe (remotely connected)

Testdisk is 1 SSD drive (intel E25), difference connection properties

Benchmark = IOMeter and 3 different scenarios:

* Sequential Read (64K blocks) on 20GB test file in MBps
* Sequential Write (64K blocks) on 20GB test file in MBps
* Random Read / Write (4K Block, 66%/33%) on 20GB test file in IOPS



# Part 3: Hardware Security part 2

## Asset Management Software

* Software to list your IT hardware (and locations)
* Very important for IT managers
* Know what you have (and what you don’t have)
* Important to keep track of potential entry points
* Microsoft SAM Checklist
* Risk Based Asset Management

## Hardware Data Recovery

Is this a problem?

* Data on the platters is held in place with very heavy electromagnetic fields
  + The magnetic force (or coercive force) is strong with this one: over 2000 oersteds
* Standard magnets have about 300 Oersteds of coervice force
  + Force dorps at even the smallest distances
* You need BIG rare-earth magnets
* The technique of removing data with electromagnetic files is **degaussing**

## Unwanted hardware failures

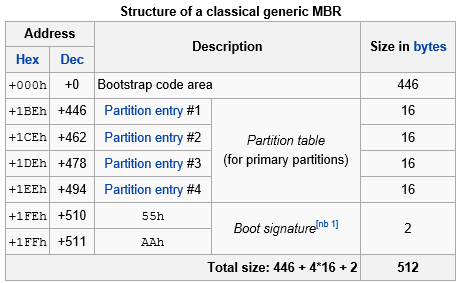
* Hard drive health check!
* Can be done using Self-Monitoring, Analysis and Reporting Technology
* Every hard drive (or SSD) supports this to report health counters
* Assists in predictable malfunctions (such as a bad sector count!)
* S.M.A.R.T. can predict 60% of HDD failures.
* S.M.A.R.T. counters have values between 1 and 253
  + Can be: internal temperature, bad sector count, Power On Hours, Spin-up Times, Read Error Rate
* Can mean different things to different vendors

# Part 4: Filesystems on Linux

## Logical structure of a hard drive

* Master Boot Record: First Sector of a Hard Drive
* Is used to boot Operating Systems
* Due to its size, there are quite some limitations

## MBR



## Cylinder Head Sector (CHS)

Formerly, sectors used to be physically addressed by their CHS address.

* Each part (C, H, S) was written in the MBR as one 8bit code (== max 256 addresses)
* First sector is 1, first heads and cylinders are 0.
* Example: if a disk has 100 sectors and 2 heads (== one platter) then cylinder 15, head 1, sector 100 was followed by cylinder 16, head 0, sector 1

This became a problem: max. 256 cylinders and 256 sectors on max. 8 heads means disks couldn’t be addressed when they were larger than about 255MB.

* There were solutions by using the 8 bits of the heads as extensions, but this was only an intermediate solutions
* NOW: CHS = 16 bits:4bits:8bits

## Logical Block Addressing

On most recent drives, CHS is no longer used, it is only there for backwards compability.

When drives are delivered from factories (unformatted) , they have been Low Level Formatted

* This means the sectors have been mapped to the bytes, usually sectors of about 512 bytes (more recent and larger HDDs: 4KB == **AdvancedFormat**)

LBA uses one address to find all data, the MBR contains the start address of each drive in 4 bytes and the number of physical sectors in again 4 bytes.

LBA is being used since IDE interfaces, 1997. The CHS addressing was only used by BIOS to address things like MBR.

When a system uses the sector addressing, CHS is ignored.

## MBR Partition Entry

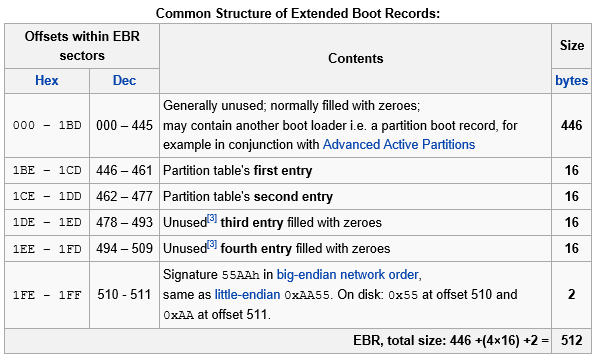
Active partition

* The bootable flag determines the active partition. Only one partition can normally be active at a time
* The active marker is used during boot: after the BIOS loads the MBR into memory and executes it, the MBR checks the partition table at its end and located the active partition.
* Then it proceeds to load the boot sector of that partition into memory and runs it.
* Each partition has a partition boot sector or volume boot record (VBR)

**Partition Types**

* Primary partition: only type that can be bootable / active
* Extended partition: special kind of primary partition that cannot be bootable or hold data but contains a link to a partition with its own partition table / boot record  
  The **EBR or extended (Partition) Boot Record**
* Logical partitions: are created within extended partitions and are thus defined in its EPBR. Can not be bootable, but there can be up to 24 logical partitions per extended partition

## EBR entry (512 bytes)



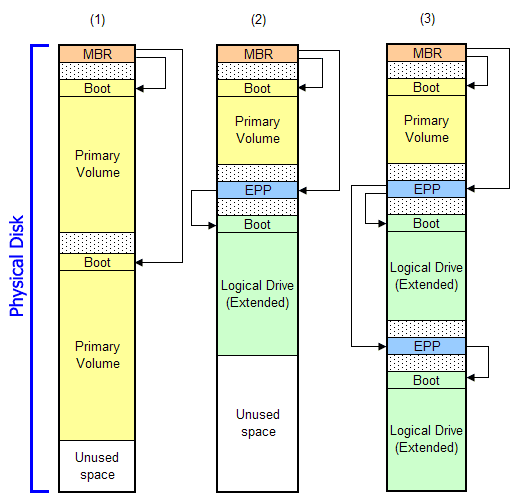
Second partition entry points to the next logical partition with its own EBR.

Can be zero if the this is the last EBR in the extended partition.

## Extended partition example

****

## Partition Layout Examples



* 1) only Primary partitions
* 2) Primary and one logical partition
* 3) Primary and multiple logical partitions

Each partition also has a boot sector which lists the file system type and other data.

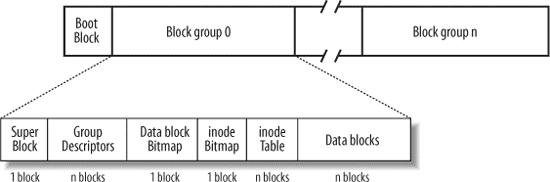
# Part 5: Filesystems

## Blocks?

Logical (filesystem) partitioning in Linux is done with blocks. These exist out of 1,4,8 or more sectors. Blocks (ext) usually have a size of 4096 bytes (== 8 sectors).

Different blocks are grouped into block groups, the first is always the super block. The use of block groups is to reduce fragmentation: the file system tries to keep blocks of the same file in the same block group, if possible.

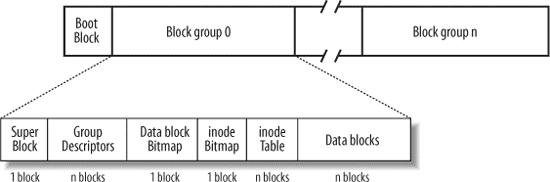
All block groups are stored sequentially, thus the kernel can address them via their index number.



## Block groups

Super block: holds the metadata of all the block, block groups and their sizes.

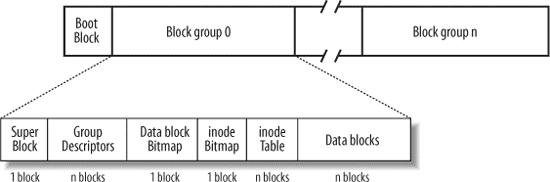
* The kernel only looks at the super block of block group 0.
* But as block groups get used, they copy the super block (and group descriptors).
* At any given time, the super block & group descriptorare thus redundantly copied.
* Can be restored with the utility e2fsck
* If a file system check gets executed (e.g. incorrect shutdown), e2fsck will copy the super block & descriptors to all used block groups
* At creation, only the first block group gets initialized (thus no backups). Therefor mke2fs initializes several backup block groups.



Block bitmap: identifies all data blocks and tags them as ‘used’ or ‘free’

* Block bitmap is always one block in size, thus there can only be 8 x b datablocks. (b is block size in bytes)
* The smaller the block groups are, the more you will need (and the larger the fragmentation)

Visualize all block groups with this tool: dumpe2fs



## A journaling filesystem?

A journal is a log (= transaction log) that captures and holds all actions on a filesystem. Specifically (successful) writes.

Introduced in ext3 (== only big difference between ext2 & ext3)

Why? 🡪 Reliability!

If ext2 (or FAT) crashes during writes, data is corrupt and entire files are possibly lost.

The OS does not know whether a file write has completed or not: filesystem check (fsck & e2fsck) is needed which can take hours.

Also: writes are not immediately stored on any drive, there is usually write caching (on-disk cache, or in OS memory). 🡪 this is why you need ‘safe-remove’ on FAT-Thumb drives.

## Filesystem reliability

Preservation

* Stable data is not affected by a crash

Predictability

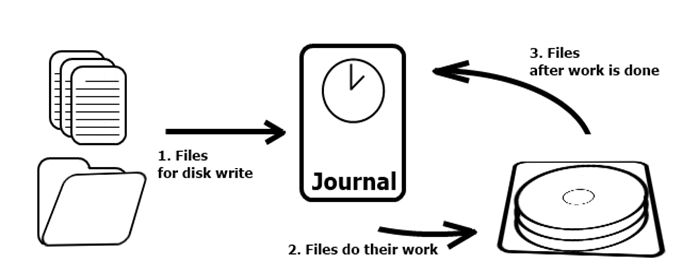
* Known failure / recovery modes for the filesystem

Atomicity

* Each operation either fully completes or is fully undone after recovery

Atomically updated

## How does it work?

The FS first tells the journal what it will do to which data.

* Metadata, disk location(s) and inode of the target file are written to the journal.

If this is completed (atomically), then the journal executes the actual write to disk (not necessarily flushed to disk).

If this is completed (atomically), the journal marks the transaction as completed.

Logging

## Logging

Old and new versions of data held on disk until the update commits.

Undo logging:

* Copy old data to the log
* Write new data to disk
* If it crashes during update, copy old data from log

Redo Logging:

* Write new data to the log
* Old data remains on disk
* If it crashes during update, copy new data from the log

## Ext3 Journaling modes

**Ordered** (default mode): Only metadata is recorded after a write is succesful. Files do not get written twice (journal & disk), so it’s faster. However, the journal is not written until the file has been written. This is secure, but it can be even more faster.

**Writeback**: only metadata is recorded, but the journal does not wait for the write to be completed. Less secure (if a write fails, the journal will not know it), but it still has timestamps for certain writes. This is the fastest type.

**Journal**: complete journaling of metadata AND actual content. Double writes happen, but security / reliability is greatest.

## Configuration

The utility “tune2fs” lists the setting of a filesystem in Linux.

* Tune2fs –l /dev/sda1 shows details about the first partition.
* Look for ‘Filesystem features’ with the ‘has\_journal’ featurue to be sure the journal is enabled.

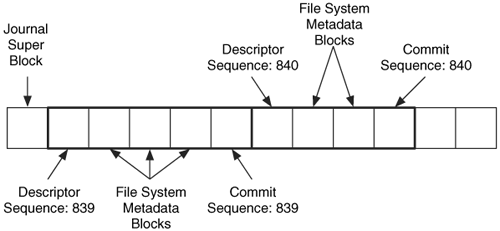
By default , the journaling mode is ‘ordered’. This can be adjusted as a mount option (default: no mount options).

As you know: these options can be configured in the /etc/fstab file:

* o data=journal
* o data=ordered
* o data=writeback

## Journal

Contains three types of data blocks

* Metadata: entire contents of a single block of filesystem metadata as updated by the transaction.
* Descriptor: describe other journal metadata blocks (where they really live on disk).
* Header: contain head and tail of the journal, sequence number, the journal is a circular structure.

A journal is circular, meaning there is a dedicated amount of space and if it’s full or on every reboot/remount, the journal starts again.

## Ext4

Each write or change results in at least two writes to disk (one to journal, one to the actual disk). This slows down some things and is problematic for some USB thumb drives or SSDs.

Ext4 still has a journal, but this can be disabled! It also has delayed allocation, allowing for drive controllers to rearrange writes and thus making them more performing.

Ext4 is also backwards compatible, it can be converted to ext3 without data loss. Ext3 can also be mounted as ext4, which results in some new features to get enabled.

If the journal itself becomes corrupt (bad sectors), the entire file system must be rescanned and potentially, massive amounts of data can be lost.

* Ext4 solves this by adding checksums to the journal but also to teach block group.

## How does this help crash recovery?

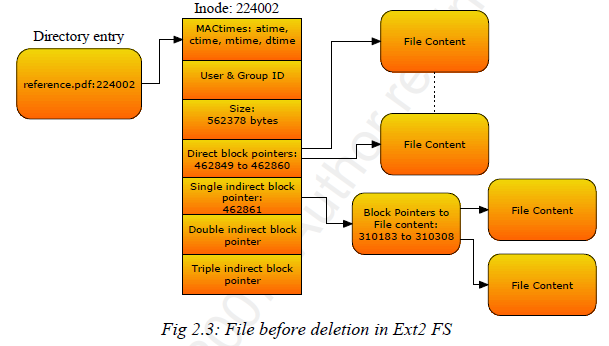
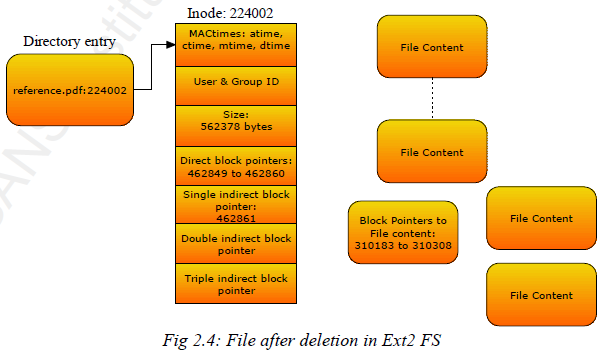
Only completed updates have been committed.

* During reboot, the recovery mechanism reapplies the committed transactions in the journal.

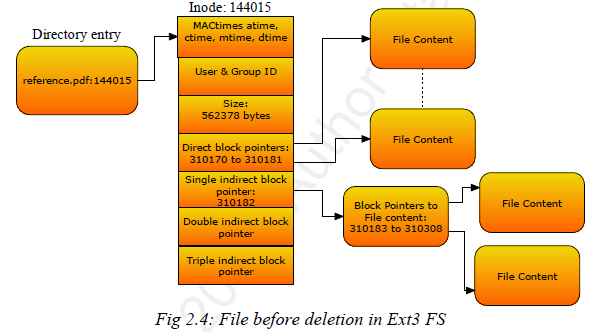
The old and updated data are each stored separately, until the commit block is written

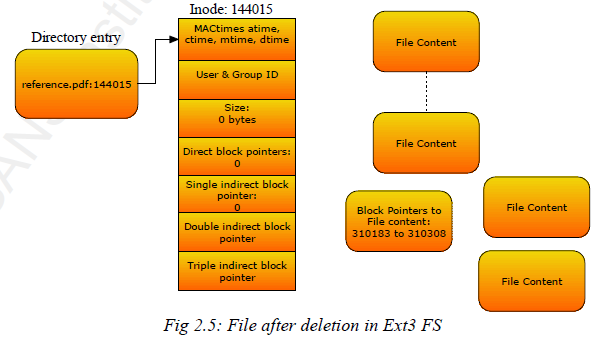
If a file is deleted:

* On ext2 (no journal): directory entry, inode and data blocks get marked empty
  + “easy” to recover



* On ext3 (journal)





## Forensic tools for data analysis (Linux)

Mount any file as read-only!

Stat: shows more information about any file, device or filesystem: links, blocks, inode, size, different timestamps

Dump2fs, tune2fs: shows or dump information about filesyste, used for filesystem configuration

Debugfs: shows contents of inodes (amongst other things),list deleted inodes (ext2!), list inode contents.

These are all standard Linux tools and can be used for (simple) file recovery on a simple file system (like ext/ext2).

## Sleuthkit

= (freeware – package manager) contains a number of tools for data analysis, journaling analysis & recovery.

* Jls: lists the journal of a give device
* Jcat: returns the content of a block in the journal
* Fsstat: shows information about ALL block groups of a filesystem
* Ils: list all inodes, present or deleted, of a device
* Istat: shows information about an inode
* Icat: returns the content of a block to which an inode points (if it exists)
* Blkls: list blocks with a certain address range

## File Allocation Table (FAT)

Very first FAT version (MS-DOS FAT) had some major limitations:

* Filenames no longer than 11 characters.
* Used 16 bit addresses = 2^16 LBA addresses \* 512 bytes = 32MB max partition size.

Filename length has later been extended to 255 characters

Later : uses the concept of clusters to address more space (1 cluster = 8 sectors)

* Default cluster-size = 4KB

## FAT12, FAT16 & FAT32

The number represents the amount of bits to keep track of the clusters.

Example: FAT12 uses a 12 bit number, 2 ^12 equals 4096 clusters (4086 usable)

Volume max. 16 MB.

FAT16: max. volume size = 2048MB (with 32KB cluster size)

FAT32: max . volume size = 8TB (with 32KB cluster size)

* Supported as of MS-DOS 7.0 and Windows 95 OSR2
* However: maximum file size = 4GB!

## exFat (or FAT64)

Developed by Microsoft as unofficial successor to FAT

Can be used if NTFS structure (journal, etc…) uses to much space.

Supported as of Vista SP1

Supported by Linux and Mac OS X (since Snow Leopard): exFAT can be used for big external drives.

Cluster size can now be 2^255 sectors (32 MB)

Maximum file size is 16 exbibites (2^64 bytes) instead of 2^32 bytes with FAT32.

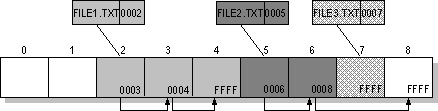
## FAT Layout

The file allocation table contains the following types of information about each cluster on the volume (see example below for FAT16):

* Unused (0x0000)
* Cluster in use by a file
* Bad Cluster (0xFFF7)
* Last cluster in a file (0xFFF8-0xFFFF)



There is no organization to the FAT folder structure, and files are given the first available location on the volume. The starting cluster number is the address of the first cluster used by the file. Each cluster contains a pointer to the next cluster in the file, or an indication (0xFFFF) that this cluster is the end of the file. These links and end of file indicators are shown below.



This illustration shows three files. The file File1.txt is a file that is large enough to use three clusters. The second file, File2.txt, is a fragmented file that also requires three clusters. A small file, File3.txt, fits completely in one cluster. In each case, the folder structure points to the first cluster of the file.

New Technology File System (NTFS)

There are three major versions:

* 1.0,1.1 & 1.2 are the old versions from NT 3.51 & NT 4
* 3.0 since Windows 2000
* 3.1 since Windows XP

However: Microsoft only has two versions, very similar to each other..

* NTFS (support for Windows NT)
* NTFS5 (no support for Windows NT, default since Win2000)

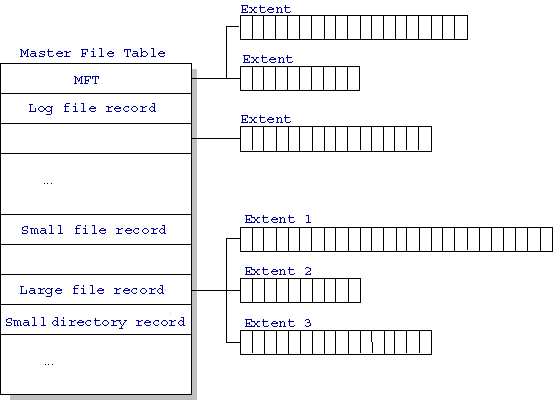
NTFS works by using clusters: bigger or smaller clusters have an impact.

Which version do you have? (needs to be run as administrator)

**Fsutil fsinfo ntfsinfo c:**

## NTFS : MFT

Each file on an NTFS volume is represented by a record (or entry) in a special file called the master file table (MFT)

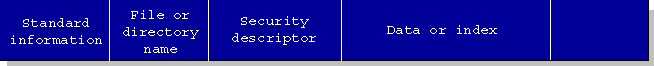
The first entry of this table describes the master file table itself, followed by a MFT mirror entry. This mirror also contains the first MFT entry.

Entries are 1KB each.

The physical location of this MFT table is always the middle of the volume 🡪 WHY? Disk performance (head looks most of the time at the middle of the disk)

## MFT records

The master file table allocates a certain amount of space for each file record. The attributes of a file are written to the allocated space in the MFT. Small files and directories (typically 512 bytes or smaller), such as the file illustrated in the next figure, can entirely be contained within the master file table record.



File access is very fast as compared to FAT:

* FAT : File Allocation Table holds names & starting addresses of the directories
* IF you need a file, the OS first reads the FAT and checks that the directory exists. Then FAT FS scans the chain of allocation units until it finds the starting address of the file.
* Then it looks through each cluster until it holds the file.

With NTFS: directory records & file records are stored in the MFT. Some mall directory records or small files (<900B) fit completely into the MFT. IF not, it contains cluster points.

## NTFS file attributes

Each file or folder has a number of attributes. This is more than only ‘hidden’, ‘read-only’ or ‘system file’ .

Elements such as the file’s name, compressed, encrypted or its security information, and even its data, are all file attributes.

If attributes can be stored completely within the MFT, they are called resident attributes.

* Attributes like filename or time stamps are always resident attributes
* A non-resident entry is stored in a cluster outside the MFT.

## NTFS System Files

NTFS includes several system files, all of which are hidden from view on the NTFS volume. A system file is used by the file system to store its metadata and to implement the file system. System files are placed on the volume by the Format Utility.

These are in a binary format, hidden for the normal user and cannot be seen using the normal explorer. Hex editors / viewer are needed for this: for example: WinHex.

Sleuthkit can also be used for NTFS detailed analysis.

## NTFS Shortcuts?

A shortcut in NTFS is just a file like any other (extension: .lnk)

Shortcuts are (by default) symbolic links: when the underlying file is removed, the shortcut will not work any longer

* Command line creation of shortcuts?
* mklink <link> <target>
* Any idea what mklink /H does ? What is the difference? Makes a hard link
* And mklink /J? Creates a directory junction ( provides the ability to create a symbolic link to a directory which then functions as an alias of that directory.)

## Symbolic vs hard vs soft links

Symbolic link == shortcut files containing a pointer to a file or directory

* Can point to directories, files or executables
* Can have its own settings (like run as administrator, attributes, security or a different icon)
* When shortcuts are opened, the application decides how to handle them (follow the “redirect” or handle the shortcut).

Junction or soft link == always on folder, but link is processed by the target folder / kernel of the running OS. Applications are unaware of the shortcut

* Even for explorer, try opening a shortcut: the address will change to that of the target folder, try the same with a junction: address will just follow the junction.
* Will not work in network paths, but will work across different volumes.

Hard links == hard links on SAME volume!

* Can be used for files or folders. Biggest difference? When target is deleted, it’s not really gone as long as there is at least one hard link.

## Multiple data streams

NTFS supports multiple data streams, where the stream name identifies a new data attribute on the file. A handle can be opened to each data stream. A data stream, then, is a unique set of file attributes. Streams have separate opportunistic locks, file locks and sizes, but common permissions.

These are also so called Alternate Data Streams(ADS), a very popular method to hide data on NTFS

ADS is the ability to fork file data into existing files without affecting their functionality, size, or display to traditional file browsing utilities like DIR or windows explorer.

Found in all versions of NTFS, ADS capabilities where originally conceived to allow for compatibility with the Macintosh Hierarchical File System ( HFS); where file information is sometimes forked into separate resources.

Most common example of ADS are details about a picture. EXIF & GEO data.

## Alternate Data Streams

Very easy to use, without the need for extra programs.

Just using type or echo with redirect (>) and colon (:).

Example:

Echo “hello” > text.txt

Echo “secret” > text.txt:stream1

More < test.txt:stream1 (printing the text)

## 10 things to know about ADS

* No limit on size of streams and there can be more than one stream linked to a normal file.
* Streams can be attached not only to files but also to folders / drives!
* Not only limited to simple text data
* No attributes of their own
* Some Browser Helper Objects (BHOs) have started storing their malicious files insides ADS
* Windows File Protection prevents the replacement of protected system files, not the ability to add streams to these files if the user has the right to write on those.
* Stream can only be executed if it’s directly called by the program.
* Internet protocols who enable file transfer (SMTP, FTP) do not support streams. But you can send files with streams over a local LAN if the target drive is in NTFS format.
* In certain cases, streams have been used to remotely exploit a web server.s

## Scenario

See the ppt dia 18.

Tools to find ADS

* Since Windows 7: DIR /R
* AdsCheck.exe
* Lads.exe
* LNS
* Ads Spy
* Hijackthis

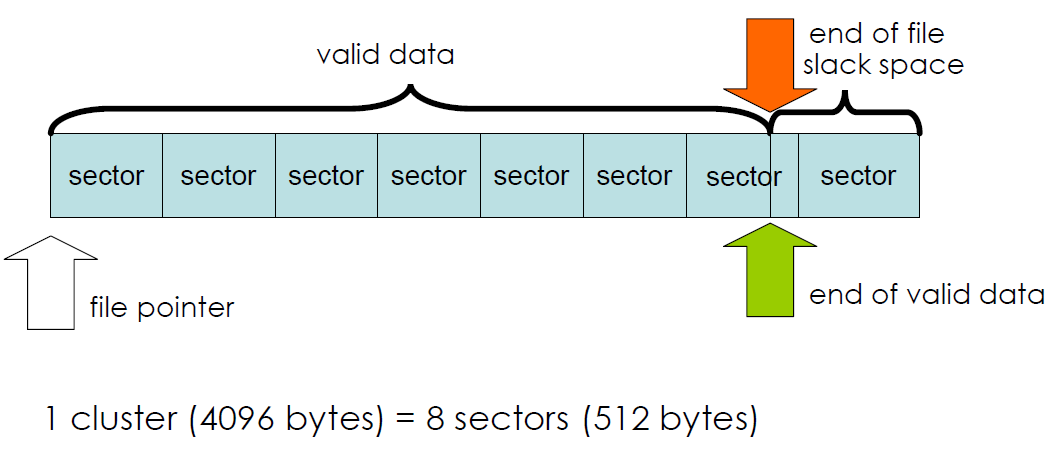
## Slack space (volume & file system)

Volume slack is the unused space between the end of a file system and end of the partition where the filesystem resides.

File system slack is the unused space in the end of a filesystem that is not allocated to any cluster.

This happens due to partition size may not be the multiple of the cluster size. For example, there are 10001 sectors in the partition, the first 10000 sectors are allocated to 2500 clusters with a cluster size of 4 sectors and the last sector left becomes the file system slack.

The size of hidden data in volume slack is unlimited as suspects can simple change the size of volume slack to hide more data. The data can be hidden in file system, however, depends on the cluster size. For example, for a file system with cluster size of 8 sectors, the maximum size of filesystem slack is 7 sectors.



## Information Concealment Engine?

An example of hiding data inside files (so this is neither Volume nor File System Slack space) is ICE.

This is a 64 bit private key block cipher (similar to DES) that (unlike DES) is designed to be secure against cryptanalysis. No weak keys etc..

An example of an application that uses this:

SNOW (= free command line tool that will add tabs & spaces to a file as “encrypted” data.

This kind of data hiding (placing data in files it does not belong) is called **STENOGRAPHY**

## File System Insertion & Subversion Technique (FIST)

* FIST’ing is inserting data into places it doesn’t belong.
* Almost stenography
* Data storage in meta-data files
* Example: journals, directory files, OLE files, …
* Modifying meta-data is dangerous
* Can be destroyed by maintenance tools (fsck, etc..)

## File System Insertion Implementations

* RuneFS
  + Stores data in “bad blocks” file
* Waffen FS
  + Stores data in the ext3 journal file
* KY FS
  + Stores data in directory files
* Data Mule FS
  + Stores data in inode reserved space

## NTFS compressed files

Windows NT / 2000 supports compression on individual files, folders and entire NTFS volumes.

Decompression occurs automatically when the file is read. The file is compressed again when it is closed or saved. Compressed files and folders have an attribute of C when viewed in Windows Explorer.

Only NTFS can read the compressed form of data. When an application such as Word or an operation system command such as copy requests access the file, the compression filter driver decompresses the file before making it available. For example, if you copy a compressed file from another compressed NTFS to a compressed folder on your hard disk, the file is decompressed when read, copied and then recompressed when saved.

NTFS allows for the compression of an entire volume, or one or more folders within a volume, or even one or more files are within a folder of an NTFS volume.

The compression algorithms in NTFS are designed to support cluster sizes of up to 4KB. When the clusters size is greater than 4 KB on a NTFS volume, none of the NTFS compression functions are available.

Each NTFS data stream contains information that indicates whether any part of the stream is compressed.

When writing a compressed file, the system reserves disk space for the uncompressed size.

## NTFS Built-in Encrypted File System (EFS)

Used to encrypt files in the background:

* It is transparent for user and any applications, there’s no risk for an user to forget to encrypt file and leave data unprotected. Once file or folder is marked as encrypted, it will encrypted in background without interaction with user. User does not need to remember password to decrypt files.
* Strong key security. In contrast to other solutions when keys are based on user entered pass phrase, EFS generate keys which are tolerant to dictionary based attacks.
* All encrypting / decrypting processes are performed in kernel mode, excluding the risk of leaving key in paging file, from where it could be possibly extracted.
* EFS provides data recovery mechanism which is valuable in business environment, giving an organization an opportunity to restore data even if the employee who encrypted it left the company.

User can invoke EFS features through Windows Explorer or by using a command line utility called chipher.exe

You can also encrypt the file or it’s whole folder: best practice is the whole folder!

Why? An application which opens the file may create temporary copies of the file while working with the document. An example is Microsoft Word. When the user opens an encrypted document, EFS decrypts it transparently for Word. Then during the work, Word creates a temporary hidden file where it automatically saves the document in the process of editing and deletes it on exit. This hidden file presents a real breach in security because it contains user data in plain (not encrypted) form. Encrypting the whole folder instead of file only, solves this problem.

Issues: see DIA 32

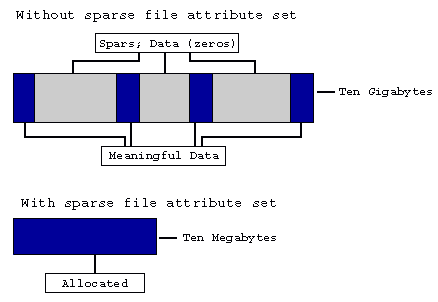
## NTFS Sparse Files (NTFS5 only)

A sparse file has an attribute that causes the I/O subsystem to allocate only meaningful (nonzero) data. Nonzero data is allocated on disk, and non-meaningful data (large strings of data composed of zeros) is not. When a sparse file is read, allocated data is returned as it was stored; non-allocated data is returned, by default, as zeros.

NTFS deallocates sparse data streams and only maintains the other data as allocated. When a program accesses a sparse file, the file system yields allocated data as actual data and deallocted data as zeros.

NTFS includes full sparse file support for both compressed and uncompressed files. NTFS handles read operations on sparse files by returning allocated data and sparse data. It is possible to read a sparse file as allocated data and a range of data without retrieving the entire data set, although NTFS returns the entire data set by default.

When the sparse file attribute set, the file system can deallocate data from anywhere in the file and, when an application calls, yield the zero data by range instead of storing and returning the actual data. File system application programming interfaces (APIs) allow for the file to be copied or backed as actual bits and sparse stream ranges. The net result is efficient file system storage and access.



Important! If you copy or move a sparse file to a FAT or a non-NTFS volume, the file is built to its originally specified size. If the required space is not available, the operation does not complete.

## Partition Corruption

* 4) Deletion of a wrong volume, it is now inaccessible by an OS and a fixmbr command will not help.
* 5) Typical boot sector virus: first 2048 bytes are wiped clearing the MBR and partition boot sector and possibly even the FAT, which is stored at the beginning
* 6) Partition chain (EBRE) Corruption, note that all logical drives are lost if first link is broken

## MBR recovery?

How does an MBR get corrupt?

* Diskpart (clean) or fdisk (d) gone wrong 🡪 only clears the MBR
* Power Loss while accessing the MBR
* Boot sector viruses
* Setting up dual boot that has gone wrong
* …

Fixmbr (or Vista’s bootrec / fixmbr) only refreshes the bootloader (first 446 bytes)

* Fixmbr is thus perfect when replacing grub with a windows boot loader or vice versa or for fixing a boat loader virus
* It will not work for other things

Vista’s bootrec / ScanOS scans all disks for partitions (recognized by Windows)

For full recovery, scan programs are needed (Bootmaster Partition Recovery, DiskPatch or Zero Assumption Recovery)

General best practice: use one or maximum two primary partitions per volume, more is seldom needed. Store the starting LBA and lengths LBAs.

* When needed, reparation can then be done manually.

## Partition boot sector recovery?

By scanning for known file systems, it is sometimes easy to recover.

Most modern file systems keep a backup of the boot sector somewhere

* FAT32 places the backup boot sector on the 6th sector of the partition
* FAT16 does not have one
* NTFS places the backup copy on the last sector of the partition volume
* With ext this boot sector is called the superblock (see later)

Backups can be restored with offline software

Example: TestDisk

## Two kinds of boot loaders

Completely written inside MBR (first 446 bytes or initial program loader)

* Mrbooter
* Lilo

Pointing to loader at a certain LBA address

* CRUB (pointing to the path /boot/grub on a partition)
* NTLDR (pointing to the file NTLDR)
* BCD (pointing to the file “Boot\BCD” on a 100MB/300MB recovery partition)

What do these boot loaders do? 🡪 Activate the correct partition and load the partition boot sector in the memory

## GPT

Often referred to as a replacement for MBR.

But with GPT: there is a MBR, which is still at LBA 0

The GPT header then begins at LBA 1, inside the MBR, the GPT partition is marked as 0xEE (partition type GPT)

Conversion can be done on the fly from MBR to GPT and back, but all volumes are lost.

Physical hard drive limit at 18 Exabyte, maximum 128 primary partitions

Each GUID partition contains a 36 character field for a readable name

## GPT (U) EFI

When using GPT with EFI, the EFI firmware does not look for the MBR on LBA0 but immediately for GPT on LBA1, skipping MBR.

GPT works with BIOS, but BIOS looks at MBR first.

The GPT header (LBA1) contains

* The partition entry locations (LBA2 & LBA-2)
* The Disk GUIDS (Global Unique Identifier or UUID in Unix)
* A 32bit cyclic redundancy check (CRC32) checksum to verify the integrity of the GPT header and the GPT.

GPT is always backed up near the end of the physical drive

## GPT Entry

The GPT partition entry begins with

* The partition type GUID (128bit) for EFI this is *{C12A7328-F81F-11D2-BA4B-00A0C93EC93B}*
* The partition GUID (128 bit) (unique for each partition)
* First and last LBA (64bit each)
* Attributes (64bit) (eg. System = 0, read-only=60, hidden=62, nomount=63j)
* Partition name (72bytes)

One GPT is usually 128 bytes, but other possibilities can be specified in the GPT header

GPT does not assume that a sector is 512 bytes.

Linux supports GPT-boot since 2008. Fdisk is not supported, use gdsik or gparted.

Windows supports GPT-boot only with (U)EFI, only with kernel 6.0.6001+ , only with x64!

## File Systems

## What is a file system?

It is a way to order files and directories in a way so that an Operating System and / or Applications and/or users can access and understand them.

In a way, a file system is a “virtualization” of the underlying logical block address structure for the above systems.

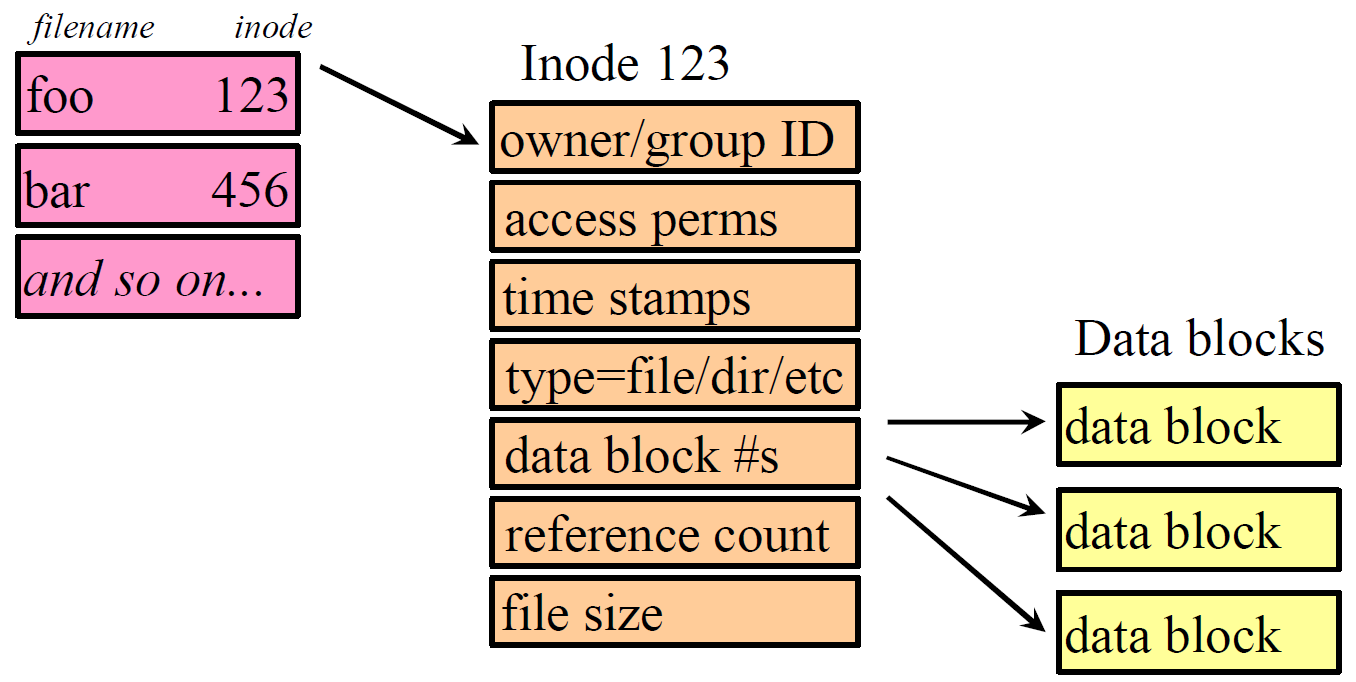
## File Systems – LINUX

Linux knows many file systems:

* Ext: Extended File System
  + 2,3,4
* Reiserfs: created by Hans Reiser from the company Namesys. Very popular druing 2006 through 2008. However support has quickly faded because Hans Reiser was charged for murder of his wife
* XFS is developed by Silicon Graphics, Inc. It is optimized for parallel IO and is created as a fully 64 bit file system.
* ZFS is created by Sun Microsystems (2005) and was the original file system for Solaris. It has been ported to Linux and has become very popular, very rapidly.   
  ZFS is a very advanced, complicated file system, combining a file system with a logical volume manager. It has build-in snapshotting, cloning, integrity verification, coaching and ZFS is a 128bit file system.

## Linux Extended File System

Designed in 1992, max file size was 2GB and max filename length was 255 characters.



## Ext problems

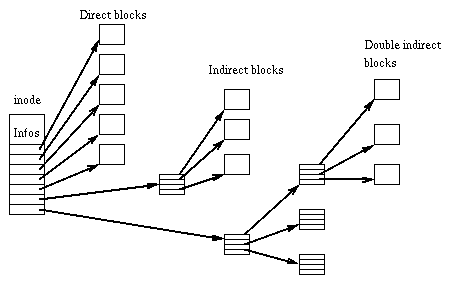
* Only one timestamp possible
* Inodes are ordered inside blocks, blocks are ordered inside superblocks using lists.  
  There was no sorting in these blocks, so addresses got fragmented. With each new file, the complete file system slowed down.
* There was need for another file system: like ext2

## Ext Features

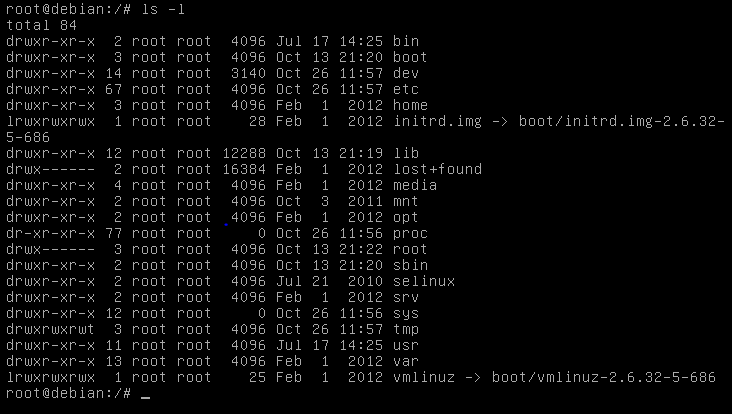
The UNIX philosophy is that everything is files

* Every device is a file, disks, keyboard, monitor, etc..
* Files that represent devices are called Device Special Files
* They are recognizable for the OS by two extra inode params:  
  A major number (which type of device) and a minor number (incremental per device type)
* There are two kind of Device Special files:
  + Character: in – and output is done one character at a time, such as a keyboard
  + Block : in – and output is done in blocks, such as a hard drive
* Links: hard-links vs. soft-links
  + Hard-links: are always within the same file system, this is how directories in ext work (files with inodes which are hard links to the files). As long as a file has at least one hard-link pointing to it, it will not be marked for deletion.
  + Soft-links are more symbolic links. It is actually a text-file containing a path.

## Ext2, the most popular Unix File System.



## Ext2 permissions



## Deleting a file on ext.

As long as there is a hard-link , the file itself will exist.

Finding the inode of a file:

**Ls –lai <file>**

Note that inodes DO NOT contain the file name, so finding filenames requires a complete file system search.

For a given inode, to find all files pointing to it:

**Find / -inum <inode>**

When all links to a file are deleted, the file is marked as “deleted”=== inode can be reused.

The only way to recover files is a scan of the complete file system.

# Part 7: Windows Analysis using the Registry

## Registry: A wealth of information

Information that can be recovered includes:

* System configuration
* Devices on the System
* User names
* Personal settings and Browser preferences
* Web browser activity
* Files opened
* Programs executed
* Passwords

## Registry History

Before the Windows Registry (DOS, Windows 3.x)

* INI files
  + SYSTEM.INI – this file controlled all the hardware of the computer system
  + WIN.INI – this file controlled all the desktop and applications on the computer system.

Individual applications also utilized their own INI files that are linked to the WIN.INI

Registry was introduced with Windows 95.s

## Definition

The Microsoft Computer Dictionary defines the registry as:

* A central hierarchical database used in the Microsoft Windows family of Operating Systems to store information necessary to configure the system for one or more users, applications and hardware devices.
* The registry contains information that Windows continually references during operation, such as profiles for each user, the applications installed on the computer and the types of documents that each can crate, property sheet settings for folders and application icons, what hardware exists on the system and the ports that are being sued.

The registry was developed to overcome the restrictions of the INI and REG.DAT files.

The registry is composed of two pieces of information:

* System-Wide information – This is the data about software and hardware settings. This information tends to apply to all users of the computer.
* User specific Information – This is data about an individual configuration. This information is specific to a user’s profile.

## Registry Organization

The Windows registry contains the following:

* Hives are utilized by the registry to store data on itself
* Hives are stored in a variety of files that are dependent on the Windows Operating system that is being utilized

The HKEY\_USERS hive contains all the actively loaded user profiles for that system.

The HKEY\_CURRENT\_CONFIG hive contains the hardware profile the system uses at start-up.

The HKEY\_CLASSES\_ROOT hive contains configuration information relating to which application is used to open various files on the system.

* HKEY\_CURRENT\_USER\Software\Classes (user-specific settings)
* HKEY\_LOCAL\_MACHINE\Software\Classes (system wide settings)

The HKEY\_CURRENT\_USER is the active, loaded user profile for the currently logged-on user

The HKEY\_LOCAL\_MACHINE hive contains a vast array of configuration information for the system, including hardware settings and software settings.

## Windows 9x Registry Files

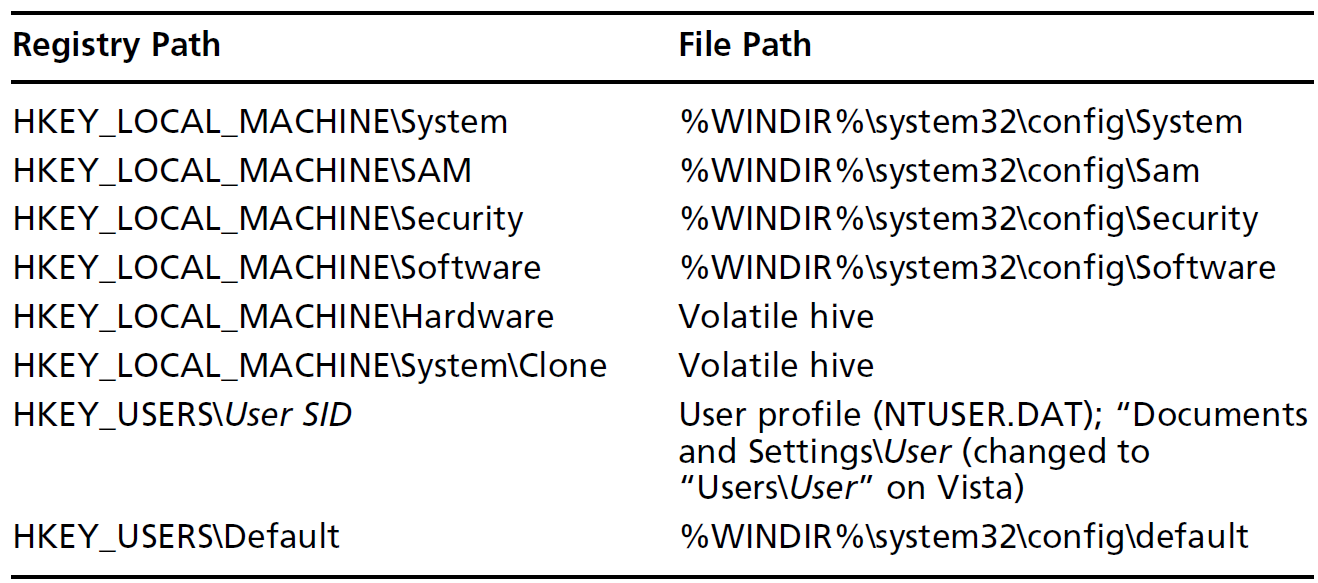
|  |  |  |
| --- | --- | --- |
| **Filename** | **Location** | **Content** |
| system.dat | C:\Windows | Protected storage area for all users  All installed programs and their settings  System settings |
| user.dat  If there are multiple user profiles, each user has an individual user.dat file in windows\profiles\user account | C:\Windows | Most Recently Used (MRU) files  User preference settings |

## Windows XP Registry Files

|  |  |  |
| --- | --- | --- |
| **Filename** | **Location** | **Content** |
| ntuser.dat  If there are multiple user profiles, each user has an individual user.dat file in windows\profiles\user account | \Documents and Settings\user account | Protected storage area for user  Most Recently Used (MRU) files  User preference settings |
| Default | \Windows\system32\config | System settings |
| SAM | \Windows\system32\config | User account management and security settings |
| Security | \Windows\system32\config | Security settings |
| Software | \Windows\system32\config | All installed programs and their settings |
| System | \Windows\system32\config | System settings |

## Windows 7/8 Registry files

|  |  |  |
| --- | --- | --- |
| **Filename** | **Location** | **Content** |
| UsrClass.dat  If there are multiple user profiles, each user has an individual UsrClass.dat file | %UserProfile%\AppData\ Local\Microsoft\Windows | Protected storage area for user  Most Recently Used (MRU) files  User preference settings |
| Default | \Windows\system32\config | System settings |
| SAM | \Windows\system32\config | User account management and security settings |
| Security | \Windows\system32\config | Security settings |
| Software | \Windows\system32\config | All installed programs and their settings |
| System | \Windows\system32\config | System settings |



Which key are loaded where, can also been seen in HKLM\System\CurrentControlSet\Control\hivelist.

Vista and Windows 7 include additional Registry hive files, specifically the Components hive file (found in the system32\config directory) and the usrclass.dat file, which is located in the C:\Users\username\AppData\Local\Microsoft\Windows directory.

The register uses cells, there are different types of cells.



Not shown here is a Security Descriptor cell.

## Windows Security and Relative ID

The Windows Registry utilizes an alphanumeric combination to uniquely identify a security principal or security group.

These are called Security IDs (SID).

There is a System SID, this is the SID for the entire system.

* This system SID is what makes a Windows Installation Unique
* Only one way to change it: **sysprep**

Each user has an unique SID

* User SIDs can be found: HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList
* Or: wmic useraccount get name.sid

The SID appears as:

S-1-5-21-927890586-3685698554-67682326-1005

## Most Recently Used (MRU)

To identify the MRU files on a suspect computer system:

* Windows 9x/Me
  + User.dat

Search should be made for MRU, LRU, Recent

* Windows NT/2000
  + Ntuser.dat

Search should be made for MRU, LRU, Recent

* Windows XP/2003/7/8
  + HKU>UserSID>Software>Microsoft>Windows>CurrentVersion>Explorer>RecentDocs

Select file extension and select item

## Registry Analysis

Perform a GUI-based live-system analysis

* Easiest, but most likely to incur changes
* Use process monitor

Perform a command-line live system analysis

* Less risky
* Use “reg” command

Remote Live system analysis

* Regedit allows access to a remote registry
* Supersan from foundstone

Offline analysis on registry files

* Encase, FTK (access data) have specialized tools
* Regedit on registry dump.
* From Windows Boot CD (just load hive inside HKEY\_USERS)

System:

* Computer name
* Dynamic disks
* Install dates
* Last user logged in
* Mounted devices
* Windows OS product key
* Registered owner
* Programs run automatically
  + Startup software
  + Good place to look for Trojans
* System’s USB devices
* Recent documents
* Recent commands entered in Windows run box

User application data:

* Adobe products
* Search terms in Google
* …

Intelliform:

* Autocomplete feature for fast form filling
* Uses values stored in the registry
  + HKEY\_CURRENT\_USER\Software\Microsoft\Protected Storage System Provider
  + Only visible on SYSTEM account
* Accessible with tools such as Windows Secret Explorer or Protected Storage Viewer

Go to

* Access Data’s Registry Quick Find Chart

## Case study: lezen dia 38

## Registry Forensics Investigation

Forensic tools allow registry investigation from image of drive.

Difference between live and offline view.

* No hardware hive
  + Dynamic key, created at boot
* No virtual keys such as HKEY\_CURRENT\_USER
  + Derived from SID key under HKEY\_USERS
  + Source file is NTUSER.DAT
* Do not confuse current and repair versions of registry files
  + %SystemRoot%\system32\config (true registry)
  + %SystemRoot%\repair (repair version of registry)

Forensic search can reveal backups of registry

* Intruders leave these behind when resetting registry in order not to damage the system.s

Software key

* Installed software
  + Registry keys are usually created with installation
  + But not deleted when program is uninstalled
  + Find them
    - Root of the software key
      * Beware of bogus names
    - HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths
    - HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall
  + If suspicious, use information from the registry to find the actual code.

Restore point

* Makes copies of important system and program files that were added since the last restore points
  + Files
    - Stored in root of RP### folder
    - Names have changed
    - File extension is unchanged
    - Name changes kept in change.log file.
  + Registry data
    - In snapchat folder
    - Names have changed, but predictably so

## User Login

According to Microsoft documentation, the start-up process for a system is not considered complete until a user logs in. When a user logs in to a system, certain registry keys are accessed and parsed so that listed applications can be run. Those keys are (in order):

* 1. HKLM\Software\Microsoft\Windows\CurrentVersion\Runonce
  2. HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run
  3. HKLM\Software\Microsoft\Windows\CurrentVersion\Run
  4. HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows\Run
  5. HKCU\Software\Microsoft\Windows\CurrentVersion\Run
  6. HKCU\Software\Microsoft\Windows\CurrentVersion\RunOnce

Each time a new users logs in to the system, keys 1,3,5 and 6 are parsed and the programs listed are run.

By default, these Run keys are ignored if the system is started in Safe Mode.

However on Windows XP and 2003 systems, if you preface the RunOnce values(keys 1 and 6) with an asterisk (\*), you can force the associated program to be run even if the system is started in Safe Mode.

## Create a log!

Unixutils can list all files in a file system and put them in a separated list for use in spreadsheets.

If an incident happened during a certain time frame, it could quickly be seen which files where accessed in that frame.

find c:\ -printf "%m;%Ax;%At;%Tx;%TT;%Cx;%CT;%V;%G;%s;%p\n"

# Part 8: Command Line tools

## Command line

* Windows (netsh)
* PowerShell
* Unix Bash
* Monitoring

Net

* net
* netsh
* netstat
* netcat
* (netplwiz)

## Net – command

Can be used for local management

net user

* user management (add, remove, password, active, …)
* On a DC: AD User Management
* Used for bulk creation/adjustment

net accounts

* user account management (account settings, pass restrictions, …)

net file

* lists open files (Windows file sharing) and closes them

Net statistics

* shows network statistics (typically CIFS)

net share

* configure local shares

net view

* lists computers in a domain / workgroup and list their shares

net use

* maps a share to folder / driveletter (net use k: [\\software.howest.be](file:///\\software.howest.be))
* lists or deletes shares

net session

* shows information about sessions

net start / stop

* starts or stops services (will be deprecated, sc is better)

net time

* configures / synchronizes NTP

## NETSTAT

= useful tool for checking network and internet connections. Some useful applications for the average PC user are considered, including checking for malware connections.

netstat –an

= keeps track of processes, PIDs, ports, IP-addresses hostnames, protocols and connection states

Interesting:

* netstat –b or faster netstat –bn

## NETSH

Network Shell

Two modes:

* interactive: just type netsh <Enter> and enter one command / line
* Non-interactive: all command can also be executed on one line:
  + for example: netsh advfirewall set allprofiles state off

netsh uses context and sub-contexts

* netsh interface – context is used for all commands for interface configuration
* netsh interface ipv6 is the sub-context for IPv6 configuration.

netsh does not use tab- or autocompletion, however all commands can be used with only the first three letters

* nets hint sh int (== netsh interface show interface ), lists interface information

netsh also has two modes: online and offline

* online (default) will immediately apply all commands
* offline, will buffer all changes and if ready a commit or flush command will apply or delete all changes (flush no longer present in windows 8)

## NETSH examples

nets hint ip reset

* will reset / repair the complete Windows TCP/IP stack
* very useful of all else fails

netsh advfirewall firewall set rule group = “remote desktop” new enable = yes

* configures the firewall to allow the pre-defined group ‘Remote Desktop’ to allow connections

netsh int ip set address name=”Local Area Connection” source=static addr=192.168.1.105

mask=255.255.255.0

netsh int ip set address name=”Local Area Connection” source=static gateway=192.168.1.1 gwmetric=1

netsh int ip set dns name=”Local Area Connection” source=static addr=192.168.1.1

netshint ip set address name=”Local Area Connection” source=dhcp

## Windows command prompt

Almost anything in GUI can be done in command prompt

Normal messages are printed to the console, with redirection they can be written to a file or device

* >> appends to file

1. means to use stdout of a command (same as >)
2. is the errout, can be redirected to a log

type: print file content

find: search for files in FS

echo: redirect a message to the current console

findstr: finds strings in a message (similar to grep in Linux)

route

…

Regshot is a good free software, usable for scripting

## Advantages / disadvantages

Advantages:

* Supports hash tables, switch statements, regular expressions, arrays, looping, conditional statements, variable scoping, pipeline, functions
* easy to discover its features
* object orientation (the output of a command is an object)
* using familiar command names
* processing text, files, registry values, XML
* creates graphical user interfaces with forms

Disadvantages

* too slow compared with unix shells
* nothing new compared with unix shells

## PowerShell Basics (°2006)

Commands

* command -parameter1 –parameter2 –parameter3

PowerShell is POSIX compliant and backwards compatible with cmd.exe

PowerShell parameter binding

* ls –Recurse; ls –rec, ls –r, ls –R (“all do the same thing”)

## Aliases

PowerShell uses a lot of aliases to link the commands.

Get-Alias

* dir, ls , gci -> Get-ChildItem
* ? 🡪 Where-Object

You can add your own aliases: Set-Alias as Get-Service

* However, these user aliases get lost with the session
  + export them (en import them each session)
  + User PowerShell Profiles, each user has a default PS profile ($Profile to show it)

## Scripting in PowerShell?

PowerShell scripts have the PS1-extension. They can be edited with any editor (notepad).

However, by default, PowerShell will NOT run scripts or configuration files.

There is an Execution Policy which, by default, presents that.

* default: RESTRICTED

If you want to run scripts, change the global PowerShell setting for the Execution Policy to (for example) Unrestricted

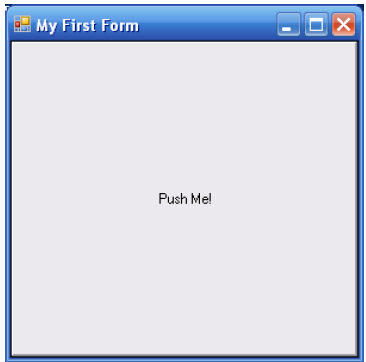
* Set-ExecutionPolicy Unrestricted

run it with ./script.ps1

## FORMS

PowerShell can load .NET libraries (assemblies) such as forms.

Example:

*[void][reflection.assembly]::LoadWithPartialName*

*("System.Windows.Forms“)*

*$form = New-Object Windows.Forms.Form*

*$form.Text = "My First Form"*

*$button = New-Object Windows.Forms.Button*

*$button.text="Push Me!"*

*$button.Dock="fill"*

*$button.add\_click({$form.close()})*

*$form.controls.add($button)*

*$form.Add\_Shown({$form.Activate()})*

*$form.ShowDialog()*

## Examples

Unix:

* $ ps -e | grep " p" | awk '{ print $1 }' | xargs kill

PowerShell

* MSH> get-process p\* | stop-process

Killing processes > 10 MB

Unix:

* $ ps -el | awk '{ if ( $6 > (1024\*10)) { print $3 } }' | grep -v PID | xargs kill

PowerShell:

* Get-Process | Where { $\_.WS -gt 10MB } | Stop-Process

## PowerShell Tools

PowerShell uses Snapins to extend its functionality.

There is many software / hardware which has Snapins for PowerShell.

* HyperV
* Active Directory
* System Center
* Storage Server

And not only by Microsoft

* Dell Management
* VMware PowerCLI
* Quest Management console

## Other tools

Default, many Microsoft Management consoles now show an extra button at the end of a Wizard “ViewScript”

* this shows the PowerShell commands to be executed at the background

Windows Server has a built-in Integrated Script Environment.

A famous OpenSource tool is PowerGUI, which has an IDE with all know commandlets, history, etc…

Another popular gimmick is called PowerGadget which uses PowerShell scripts to steer gadgets and GUI gauges.

## Linux Bash

Linux has no notion of extensions, making a file (script) executable is done by configuring it’s rights (mode).

* chmod +x myscript or chmod 700 myscript

They are a lot of script-languages in Linux (interpreters), most are built in:

* sh
* bash
* ksh
* csh
* perl
* ruby
* python
* php

## Linux Script

Linux Script which are “well-behaved” must start with the so-calls shebang:

* #! /bin/bash
* this tells the environment which interpreter to be used.

Scripts can be added as a startup service by copying them to the folder /etc/init.d and running this command to make them auto start

* update-rc.d myscript defaults

# Part 9: Domain Name Service (DNS)

## Terminology

Zone

* this is a “domain”: a collection of hostnames /IP’s all managed together
* Example: Howest.be is a zone, www, start, leho, software etc.. are all part of the Howest.be zone.
* two kinds of zones: forward & reserve zones
  + forward = name to IP
  + reserve = IP to name

Nameserver

* this is the server software that answers any DNS question
* Either from cache or by forwarding the request

Authoritative Nameserver

* Nameserver that is responsible for a certain zone
* Usually one Nameserver is the zone master. It contains the IP’s & names for that zone. Sometimes there are masters & slaves that are all authoritative for a zone.

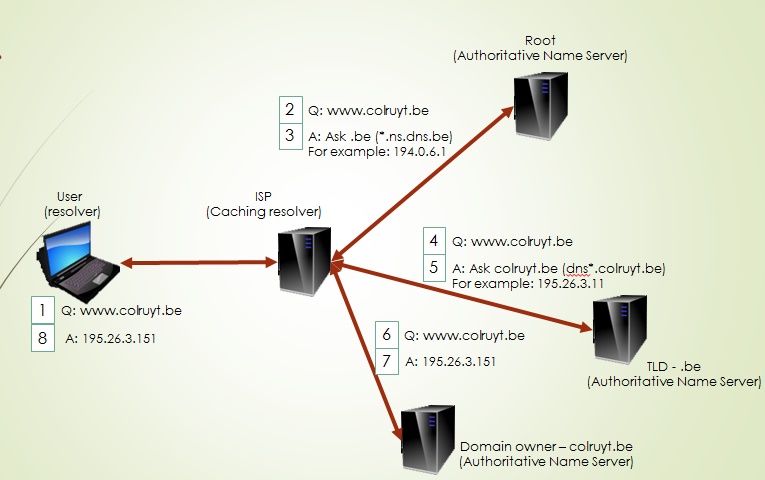
Resolver

* This is the DNS client that ask the questions
* In Linux / UNIX systems, this servers-to-ask is found in /etc/resolve.conf
* In Windows, it is just part of the Network connections setup.

Records

* there are different record types, also called resource records
* mostly we want to know the mapping of a name to an IP(v4) address
  + A record
* but we can also ask for other types of records
  + AAAA = IPv6 address
  + MX = mail exchanger
  + NS = name server
  + SOA = start of authority
  + CNAME = aliases for a name
  + ….

## How does it work?



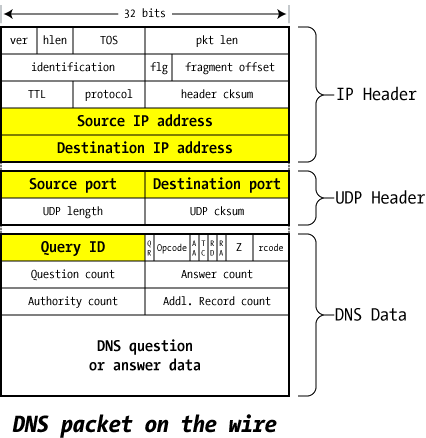
So it comes all down to the root servers

These names and IP addresses are hard coded into each / most DNS server

The Nameserver picks one at random and sends the query to that server.

* these servers do not have A records
* only referrals to other servers: a list of TLD servers for that domain

These DNS-queries happen trillions of times a day and very fast. DNS is built upon the UDP protocol (port 53) !



Also: there is not ONE machine that knows everything.

## How does a DNS packet look like?

the Query ID: unique ID created by the query packet (DNS server).

* Each query that belongs to that query must have this ID
* A nameserver can have multiple outstanding queries at a time, so this ID helps identify the queries and responses
* Also called Transaction ID

Question count: identified what we are looking for : the query type (A), the name ([www.colruyt.be](http://www.colruyt.be)) and a class (IN=Internet)

## Issues with DNS

* one of the oldest protocols used on the internet
* created in the time where all hosts where known and trusted
* design of DNS is inherently insecure, security was not a requirement

## Attack Vectors

Channel

* Communication between master en slave servers
* Communication between resolvers

Data (in caches)

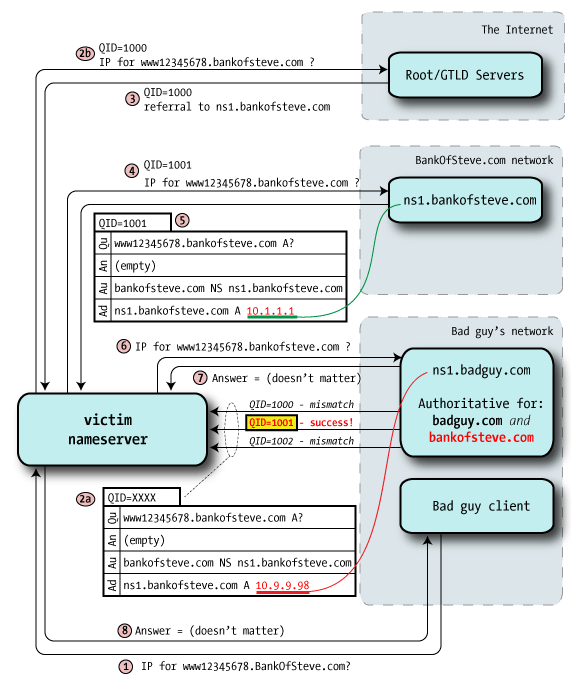
* cache poisoning
* Man-in-the-middle
* Data modification
* Kaminsky attack

Remains in cache until TTL expires, attackers use very long TTL’s.

## The Time-To-Live (TTL)

Each DNS query that is executed by a DNS server (or resolver!!!!) gets cached locally.

However: it may be stored indefinitely. The data may become stale (outdate) and therefore incorrect.

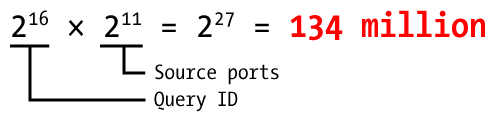


That’s why each record has a TTL; the maximum amount of time a DNS “client” may keep the records in cache. It is measured in seconds.

## Attack example

* By flooding a nameserver with answers to queries that may not have been asked. Sooner or later there will be an answer that corresponds to a correct DNS query.
* the “forged” answer contains the link to a BAD name server which gets cached in the ISP nameserver for a TTL that was specified in the DNS answer (e.g. 2 years).

## Kaminksy attack

* DNS uses a 16 bit transaction ID (65536 possible ID’s) what makes it easy to guess.
* TTL was believed to prevent an exploit of this property (caching)
* the attack uses related domains that don’t exists and do not have TTL and hijack the NS records, not the A records
* Temporary solution is to use source port randomization to increase size of the Transaction ID.

## However?

* Local poisoning still possible
* Sending DNS replies to local clients can still be done very fast
* DNS traffic can also be sniffed, intercepted and altered

## Better solution?

DNSSEC

* created to secure for the data part
* other solutions exists for the channel part (like VPN tunnels, TSIG, …)

First version was in 1997, current version from 2005.

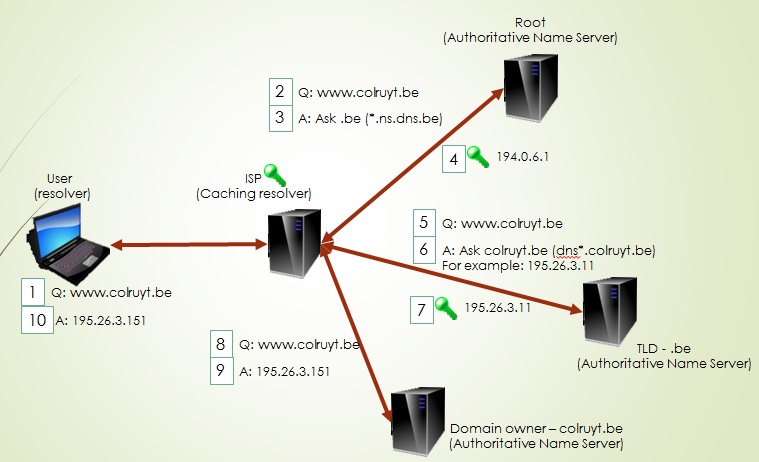
Designed to be backwards compatible.

All zone elements are signed (not encrypted) using Asymmetric Cryptography.

Publish the public part of the signing-key in the zone above you

🡪 this creates a chain-of-trust just like regular DNS

## How does DNSSEC work



## Forensic use

In a local machine, let’s consider three layers of DNS resolving:

* the host file is checked (on both Windows & Linux!)
* Local DNS
* Forwarding DNS. Such as public ISP DNS servers, usually caching DNS servers

Each of these layers can be attacked.

## DNS cache

Every operating system also has a DNS cache (that is queried before checking the hosts-file)

* each time Windows resolves a FQDN, it stores it’s IP address in the local DNS cache
* Can been seen with:
  + ipconfig /displaydns
* And can be cleared with:
  + ipconfig /flushdns

This way, it is always possible to detect what sites the user has been visiting

* even if the browser is set to private browsing or you clear the local history

It is sometimes even possible to not only find which hosts the serve has visited, but also when he has done that (for the first time).

Each DNS result has a default TTL . If we look at the remaining TTL in the ipconfig /displaydns command and we combine that with the default TTL we can calculate when the resolving has happened and therefor when the site was visited.

Example:

nslookup -type=A -debug [DN] [authoritative NS] | findstr /i ttl  
nslookup -type=A -debug www.colruyt.be 195.26.3.11 | findstr /i ttl

*ttl = 3600 (1 hour)*

## DNS cache clear?

The cache is only kept in System Memory

The DNS cache is cleared on multiple occasions

* every reboot / standby
* Every time you switch the network

There is also a registry key to disable local cache

* HKLM\SYSTEM\CurrentControlSet\services\Dnscache\Parameters
* Create DWORD value “MaxCacheEntryTTLLimit” where you can configure a static TTL value for every DN. When set to 0 🡺 no caching!