Operating System & Application Forensics

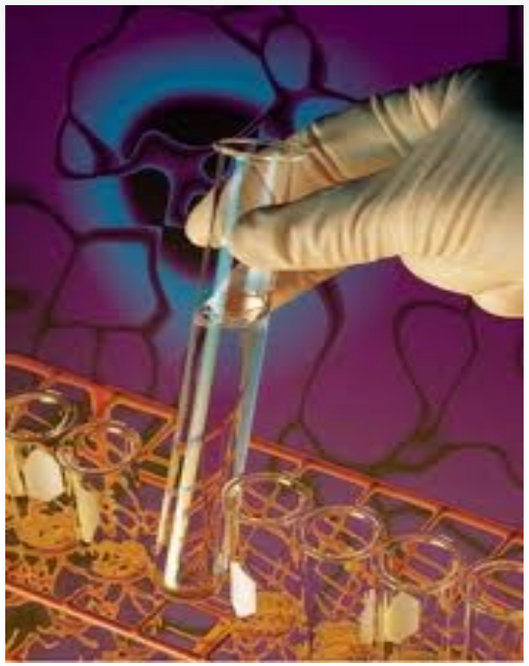
Davide Balzarotti
davide@iseclab.org

I Extracted the Data. Now What?

- It depends on the investigation
  - Look for particular artifacts
    (child pornography, data exfiltration, credit card numbers, …)
  - Look for signs of compromise
    (malware, logs about unusual activity, …)
  - Reconstruct a user activity in a certain time frame
    (computer used to commit a crime)
  - Gather information about the suspect
    (IRC contacts, emails, phone numbers, visited webpages, ...)

Summary

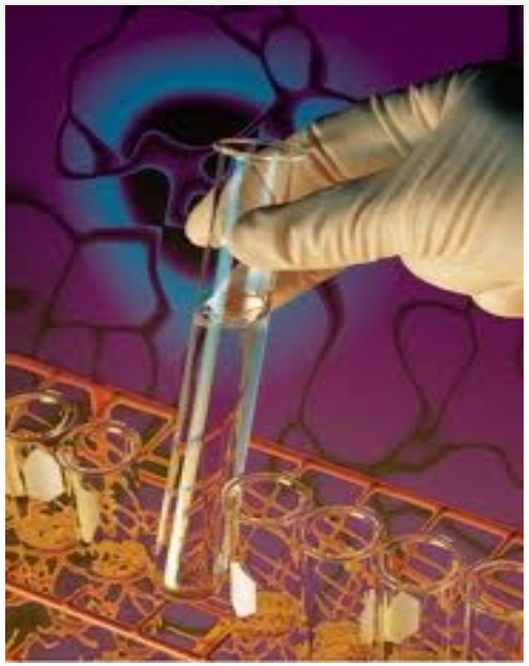
- OS-independent
- Linux Artifacts
- Windows Artifacts



OS-independent

Summary

- OS-independent
  - Files (type, camouflaged, timestamps …)
  - Document Metadata
  - Timelines
  - Web Browser Forensic
- Linux Artifacts
- Windows Artifacts



Files

- A sequence of bytes used as *data container*
  - Logically (but not necessarily physically) consecutive
- File metadata (name, path, permissions, timestamps...) are not stored in the file, but in the filesystem
- The file type determines the format of the file (what each byte means and how it has to be interpreted)
  - Some are well structured (e.g., a `tgz` archive)
  - Some are somehow structured (`html`, `tex`, ...)
  - Some are not structured at all (a `text` file)
- Where is the file type stored? – NOWHERE –

File Type Identification

- Encoded in the file extension (Windows approach)
  - Does not work if the filename is no longer available
  - Can be easily changed

File Type Identification

- Encoded in the file extension (Windows approach)
  - Does not work if the filename is no longer available
  - Can be easily changed
- Determined by looking at the header/footer/... (Unix Approach)
  - Based on syntactic rules on the file content
  - Difficult to apply to unstructured files
  - Sometimes can be fooled by adding fake byte sequences

File Type Identification

- Encoded in the file extension (Windows approach)
  - Does not work if the filename is no longer available
  - Can be easily changed
- Determined by looking at the header/footer/... (Unix Approach)
  - Based on syntactic rules on the file content
  - Difficult to apply to unstructured files
  - Sometimes can be fooled by adding fake byte sequences
- Guessed by looking at the content
  - Can be applied when only a fragment of the file is available
  - Statistical analysis (byte distribution and correlation)
  - Research topic in computer forensic

File Type Identification - Tools

- `libmagic` (used by the `file` command) is the standard in *nix systems
  - It searches for *magic patterns* defined in a configuration file (typically under `/usr/share/misc/magic`)
- Other databases of file signatures exist (e.g., `http://www.garykessler.net/library/file_sigs.html`)
- Question: what happens if you concatenate two files?

File Type Identification - Tools

- `libmagic` (used by the `file` command) is the standard in *nix systems
  - It searches for *magic patterns* defined in a configuration file (typically under `/usr/share/misc/magic`)
- Other databases of file signatures exist (e.g., `http://www.garykessler.net/library/file_sigs.html`)
- Question: what happens if you concatenate two files?
  - Many file formats (jpg, zip, doc, mp3, elf, …) do not care if extra data is appended at the end of the file
  - The first file can be used normally, while the second becomes … *invisible*

Camouflaged Files

- Very simple way to hide information by changing the file extension
  - E.g., disguise a `jpeg` picture by calling it `poker.exe`
  - In Linux, the file would keep “working” also with the wrong extension (e.g., attacker often upload perl script renamed as `png` files)

- Analysis
  - Run file on each file and compare with the extension to detect mismatch
  - `sorter` (from sleuthkit) can do the analysis for you on a disk image
    - Limited number of extension supported

```
$ cat /usr/share/tsk3/sorter/default.sort | grep "^ext"
```

MAC Time

- Timestamps have different resolutions
  - Ext2/3 – number of seconds (nanoseconds in ext4) since 1/1/1970
  - NTFS – numbers 100-nanosecond since 1/1/1601
  - FAT32 – from 1/1/1980 (resolution from 1/10 to 2 sec)

- Keep in mind:
  - Time is stored in UTC in certain filesystems (NTFS) or in local time in others (FAT)
  - NTFS and EXT4 also store the creation time, ext2/3 don't
  - Vista does no longer track last access time by default
  - Starting from the kernel 2.6.30 (middle 2009), the default Linux behavior is to update `atime` <u>only</u> if at least one of the following conditions is true:

    (`atime` < `mtime`) or (`atime` < `ctime`) `or` (`atime` is 24h older)

Accessing `crtime` in ext4

- BSD is able to retrieve the creation time but the Linux `stat` interface does not support it
  - ~~They are waiting the new `statx()` syscall that has been in development since forever :(~~
  - After over 6 years (!!) statx has been merged in the kernel in March 2017 and it is available from kernel 4.11

- In the meantime, you can print the creation time in `ext4` using:

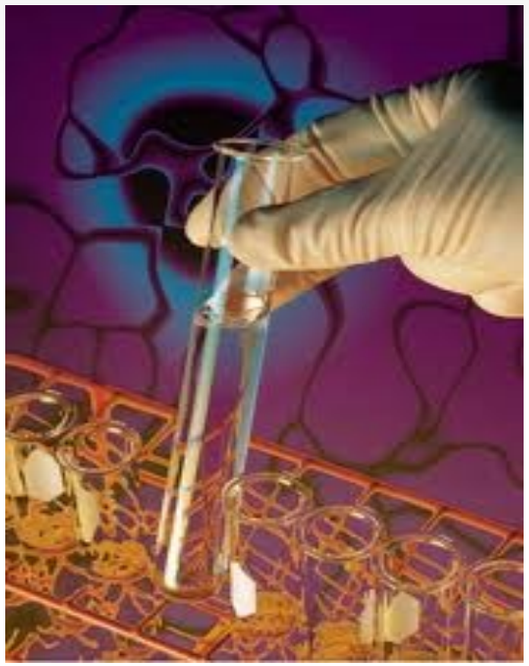
```
$ debugfs -R "stat <21757972>" /dev/sda
```

inode

device

Summary

- OS-independent
  - ✔ Files (type, camouflaged, timestamps, …)
  - Document Metadata
  - Timelines
  - Web Browser Forensic
- Linux Artifacts
- Windows Artifacts



Document Metadata

- Metadata stored in various document formats is often more important that the document itself
  - It can store timestamps, author names, GPS coordinates, software versions, old revisions of the documents, OS versions, complete file path, …
- `extract` is a generic tool that can extract metadata from several file formats
- Ad-hoc tools do a better job on specific files
  - In particular, it is worth examining in details at least pictures, MS Office documents, and PDF files

Document Metadata

- Images

```
$ exiv2 -pa <filename>
```

  - Can print several pages of information

Digital Ballistic tries to associate a picture to the device (or the software) that generated it

- Without using metadata
- E.g., "*Digital Ballistics with Calvin*" proposes the use of JPEG quantization tables to identify the device

Document Metadata

- PDF Documents
  - `pdfinfo` (part of xpdf package) – generic metadata
  - `pdfresurrect` – retrieves previous versions that have changes appended with incremental updates
  - `dumppdf` (part of pdfminer) – extract the content of pdf streams
- Office documents
  - Microsoft OLE (`doc, docx`)
    - `wvSummary` (part of the wv package)
    - `wmd.pl` – word metadata dumper by H. Carvey
  - Openoffice xml (`ooxml`), Opendocument format (`ODF`)
    - Plain zip archives containing xml documents and binaries

Timeline

- A way to organize the collected information by indexing and ordering each entry according to its timestamp
  - Collection: events and their associated temporal data are gathered and saved in a body file (typically in `csv` format)
  - Analysis: the events are ordered and displayed to the user in a convenient way (tables, graphs, …)

- Originally focused on a single data source, e.g. the filesystem
  - Extract the file metadata and store them in `.csv` format

```
$ fls  -m "/" -r xxx > body      (for disk images)
$ macrobber / > body             (for live systems)
```

  - Show the timeline

```
$ mactime -b body -d
```

Timeline

```
Thu Jan 01 1970 01:00:00,2285,...b,  -rw-r--r--,1000,1000,0, "forensics/projects/net1.txt"
….
Thu Mar 08 2012 15:09:35,2285,m.c.,-rw-r--r--,1000,1000,0,"forensics/projects/net1.txt"
Thu Mar 08 2012 15:09:44,2285,.a..  ,-rw-r--r--,1000,1000,0,"forensics/projects/net1.txt"
….
```

Timeline

Thu Jan 01 1970 01:00:00,2285,...b, -rw-r--r--,1000,1000,0, "forensics/projects/net1.txt"
....
Thu Mar 08 2012 15:09:35,2285,m.c.,-rw-r--r--,1000,1000,0,"forensics/projects/net1.txt"
Thu Mar 08 2012 15:09:44,2285,.a.. ,-rw-r--r--,1000,1000,0,"forensics/projects/net1.txt"
....

Timestamp

Action [macb]

```
m = modification
a = access
c = change
b = creation
```

Super-Timeline

- Traditional filesystem timelines cover only a small part of the available evidence
- To get a better overview of all the events that took place we need to incorporate other data sources in the timeline
  - The result is called a super-timeline
- log2timeline (part of the Plaso framework)
  - Framework containing many independent modules to extract time-based information from different sources
  - It supports several backends that save the timeline in different formats

logs2timeline

```
$ log2timeline -f <format> -z <tzone> -o <output> file
```

- format: currently supports over 100 data sources, organized in 7 categories (run `-f list` to list them)
- tzone: timezone that was used on the computer that the log files belonged to
- output: select the output format (default is csv)

```
$ timescanner -d directory
```

- recursively scans through a directory and extract data from the files that `log2timeline` supports
- supports similar options of `log2timeline` (output format, timezone, ...)

Timeline Analysis

- The super-timeline of an average laptop can contain millions of entries... therefore it is easy to miss the relevant info
- Data reduction / Locality analysis
  - Focus the time window around the incident time
  - Focus the analysis to the proximity of a known event
- Look for anomalies
  - Activities in abnormal day time
  - Modification to system directories
  - Lack of modifications when you expect them (antiforensic?)

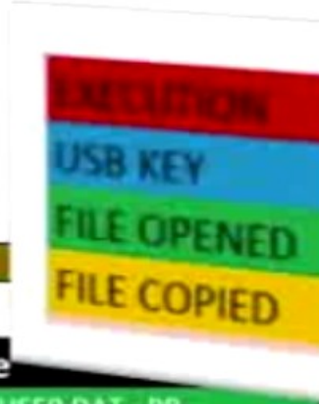
Timeline Analysis

- Grouping
  - Create collective events that belong to the same type
- Event Correlation
  - Create new event from the correlation of different entries that belong to the same action
- Visualization
  - Spreadsheet
  - Logs visualization tools
    - Elastic Search + Kibana

Important

- Timelines just give you an ordered list of *low level* events
- It is up to you to correctly interpret those events, formulate hypothesis, and draw conclusions
- E.g.,
  - Which MACB times are changed when you copy a file?
  - Which MACB times are modified when you extract a file from an archive?

Final Timeline – Our Goal



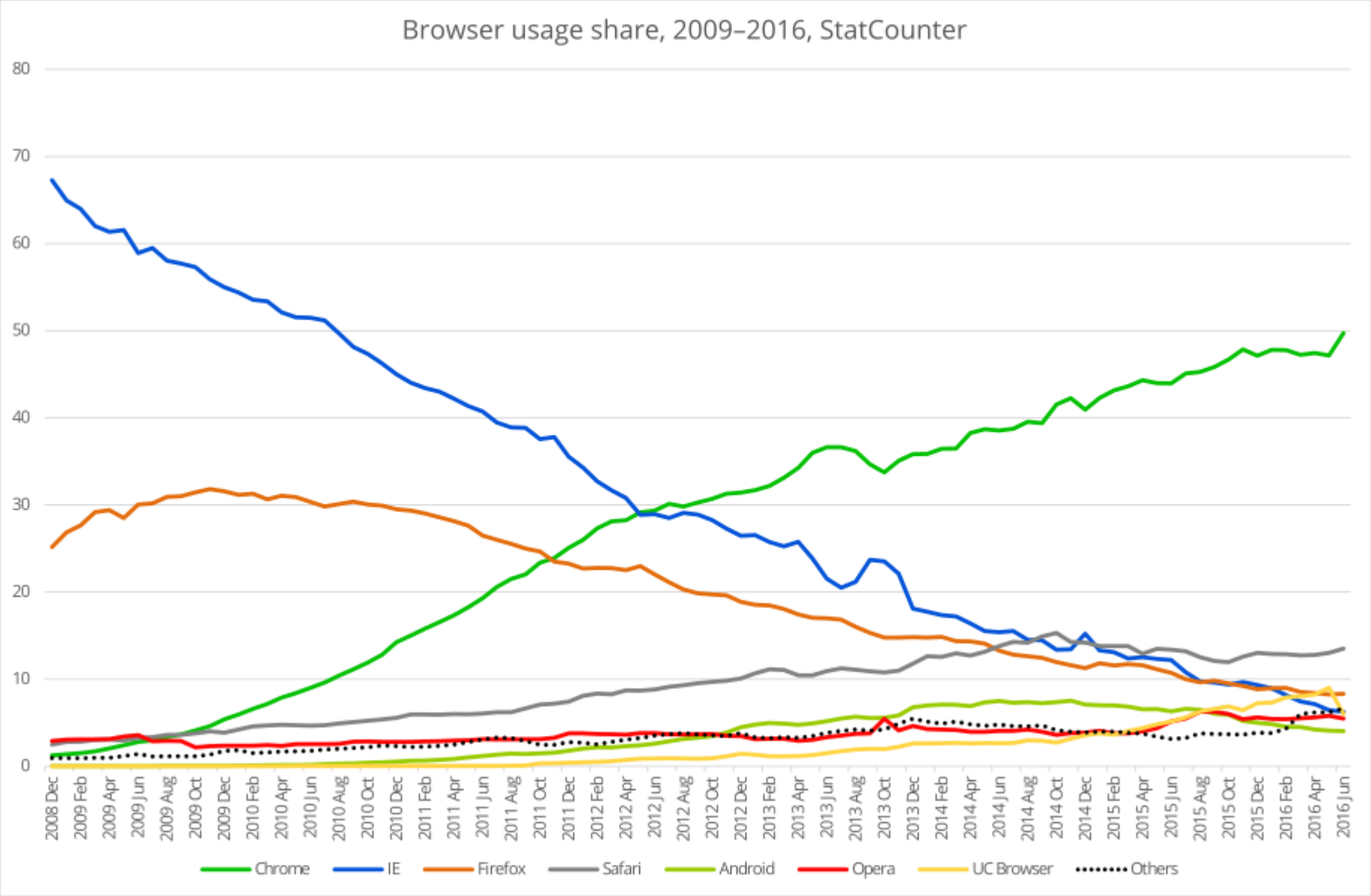
Date (Newest to Oldest)	Artifact Involved	Action	Source
1/16/2009 23:27:02	F:\TIVO Research - CONFIDENTIAL - BACKUP2.doc	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS, History INDEX.DAT
[illegible]	[illegible]	[illegible]	[illegible]
1/16/2009 23:26:50	F:\TIVO Research - CONFIDENTIAL - BACKUP2.doc	Copied to F:\ (WORKOUT IPO)	LNK FILE - Target Creation Time
1/16/2009 23:25:30	F:\SECRET\SECRET.zip	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS, History INDEX.DAT
1/16/2009 23:25:30	F:\SECRET	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS
[illegible]	[illegible]	[illegible]	[illegible]
1/16/2009 23:25:13	F:\SECRET\SECRET.zip	Copied to F:\ (WORKOUT IPO)	LNK File - Target Creation Time
1/16/2009 23:24:06	(F:) USB KEY (Apple iPod)-- iserial# - 000A270010C4EB6E	Key Inserted	SETUPAPI.LOG - USBDEVFORENSICS
1/16/2009 23:20:59	E:\CONFIDENTIAL_SPREADSHEETS.zip	Last Opened	LNK File - Modification Time, History INDEX.DAT
1/16/2009 23:20:59	E:\ (Dblake Personal) folder	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS
1/16/2009 23:18:30	E:\TIVO Research - CONFIDENTIAL.doc	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS, History INDEX.DAT
1/16/2009 23:18:26	E:\Blue Harvest Business Plan v1.doc	Last Opened	LNK File - Modification Time, NTUSER.DAT - RR - RECENTDOCS, History INDEX.DAT
1/16/2009 23:18:26	E:\ (Dblake Personal) folder	First Opened	LNK File - Creation Time
1/16/2009 23:18:19	E:\TIVO Research - CONFIDENTIAL.doc	Copied to E:\ (Dblake Personal)	LNK File - Target Creation Time
1/16/2009 23:18:15	E:\CONFIDENTIAL_SPREADSHEETS.zip	Copied to E:\ (Dblake Personal)	LNK File - Target Creation Time
1/16/2009 23:18:10	E:\Blue Harvest Business Plan v1.doc	Copied to E:\ (Dblake Personal)	LNK File - Target Creation Time
1/16/2009 23:15:20	(E:) USB KEY (Dell_Memory_Key)-- iserial# - 086086412140E1C2	Key Inserted	SETUPAPI.LOG - USBDEVFORENSICS

From SANS Super Timeline Analysis presentation

Web Browser Forensics

- Web browsers store plenty of data about the users behavior
  - Browser history
  - Browser cache
  - Cookies
  - Form information (from form auto-completion)
  - Bookmarks
- Open source tools are available to extract most of the data

Web Browsers



Chrome

- Files location
  - XP: `\Local Settings\Application Data\Google\Chrome`
  - Vista/7/8/10: `\AppData\Local\Google\Chrome`
  - MacOSX: `~/Library/Application Support/Google/Chrome/`
  - Linux: `/home/$USER/.config/google-chrome/Default/`
    `/home/$USER/.config/chromium/Default/`
- Uses a number of `sqlite3` databases to store most of the info
  - Simple database-in-a-file
  - Play with it with `sqlite3` or `sqliteman`
  - Deleted rows remain in the file until they get overwritten

Chrome

- Interesting tables
  - keyword_search_terms
  - download
    - Timestamps in seconds since January 1, 1970 UTC
  - urls
  - visits
    - Timestamps in microseconds since January 1, 1601 UTC
    - Saves the reason why each url was retrieved
      (AUTO_SUBFRAME, LINK, TYPED, FORM_SUBMIT, ...)

Other Interesting Files

- Known format
  - Web data – (sqlite) form autofill data
  - Cookies – (sqlite) those little sweet things
  - Bookmarks – (json) list of favorites
  - History index – (sqlite) content of the pages used to index words
- Binary format
  - Visited Links – Used to color the visited links
  - Last Tabs – Used in case of crash to restore the open tabs
  - Last Session – Used in case of crash to restore the session
- Cache
  - http://www.chromium.org/developers/design-documents/network-stack/disk-cache

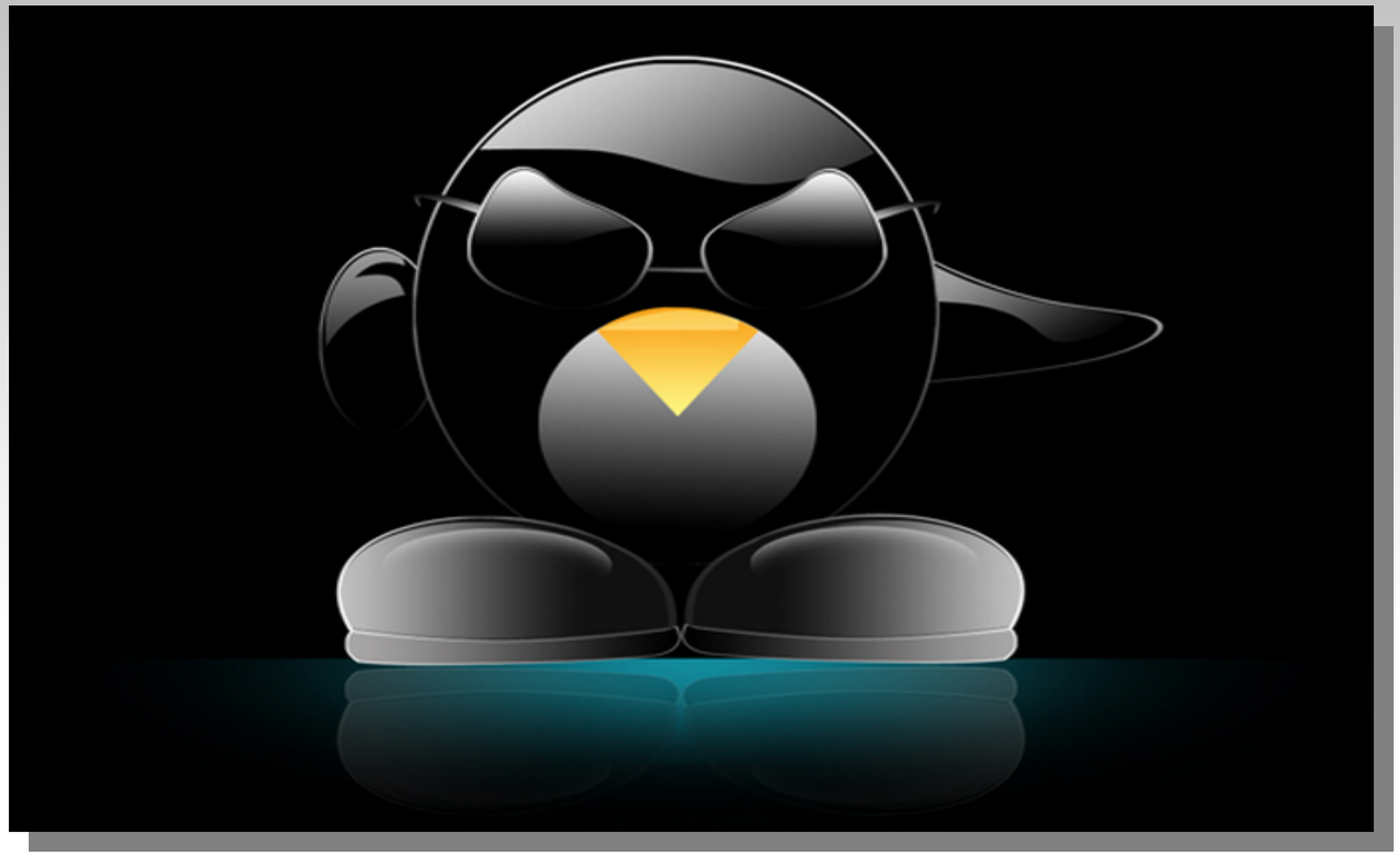
Cookies

- Cookies sometime store additional informations that may not be available in the logs / network traces

- E.g.: Google Analytics
  - Used to collect statistical information about a website visitors
  - Documented structure:

__utma: <domain hash>.<visitor ID>.<first visit>.<previous>.<last>.<# of sessions>

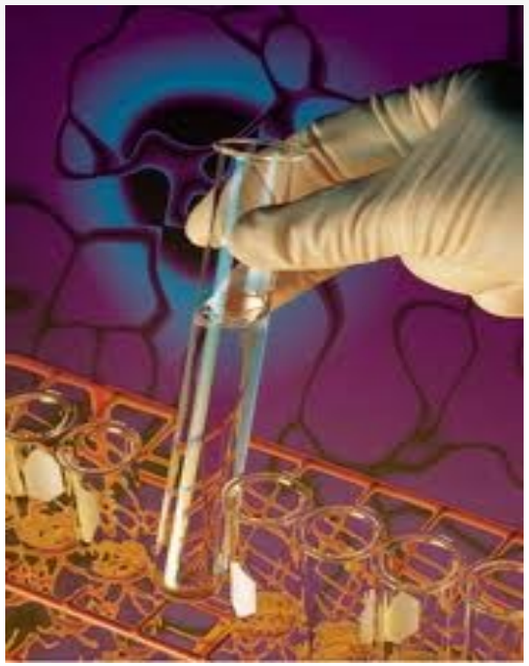
__utmz: <domain hash>.<last time>.<sessions>.<sources>.<variables>



The Linux World

Summary

- ✔ OS-independent
- Linux Artifacts
  - General Information
  - User accounts
  - Shells and SSH
  - Log files
- Windows Artifacts



In a Nutshell

- No central repository of system and user information
  - But everything follows well defined rules.. no need to reverse engineer data structure
- Most of the configurations and log files are in plain text
  - Easy to analyze
  - Easy to grep
- Often installed on a `ext4` filesystem
  - Mey be hard to recover undeleted files

System Configuration

- `/etc/` contains separate files and/or sub-directories containing the configuration of each application (in plain text)
  - Computer name
    - `/etc/hostname`
  - Release name and version
    - `/etc/*-release`
    - `/etc/issue`
  - Timezone
    - `/etc/sysconfig/clock` (red hat)
    - `$ zdump /etc/localtime` (debian)
  - Kernel image
    - Look under `/boot/`

User Accounts

- Accounts information are stored in the `/etc/passwd` file
  - Permissions are associated to numeric user and group identifiers (UID and GID)
    - UID = 0 means the user has superuser (ROOT) privileges
  - For each user, the file contains the path of the home directory
  - Home directories are often left behind when accounts are deleted
- Password hashes are stored in `/etc/shadow`
  - In the format: `$id$salt$hashed`
  - Bruteforce with *John the Ripper*
- Group memberships are stored in `/etc/groups`
- Check `/etc/sudoers` for account with (limited) root privileges

SSH

- Personal user configuration are stored in `~/.ssh/config`
- All the machines a user ever connected to are saved in `~/.ssh/known_hosts`
  - SSH version 4 and later is normally configured to store only an hash of the machine name
  - You can bruteforce the entries if you know more or less what you are looking for (e.g., other hosts in the local network)
  - A little perl script is available to do that:

```
$ known_hosts_bruteforcer.pl -i -s 193.55.114.0
```

- Check for entries in `authorized_keys` for unauthorized access

Command History

- Shell history saved in `~/.bash_history`
  - By default, it does not contain timestamps
  - Can be freely edited and/or deleted by the user
- The SUDO command history is saved in `/var/log/auth.log`
  - Protected from normal users, but if one has sudo access...
- SSH does *not* save a log of the commands that are executed

Persistence Mechanisms

- Start-up scripts
  - `/etc/inittab`
  - `/etc/rc.d/*`
  - `/etc/systemd/`
  - `/etc/init/`
- Internet super-server daemon
  - `/etc/inetd.conf` or `/etc/xinetd.conf`
- Cron jobs
  - `/etc/cron*`
  - `/var/spool/cron/*`

Log Files

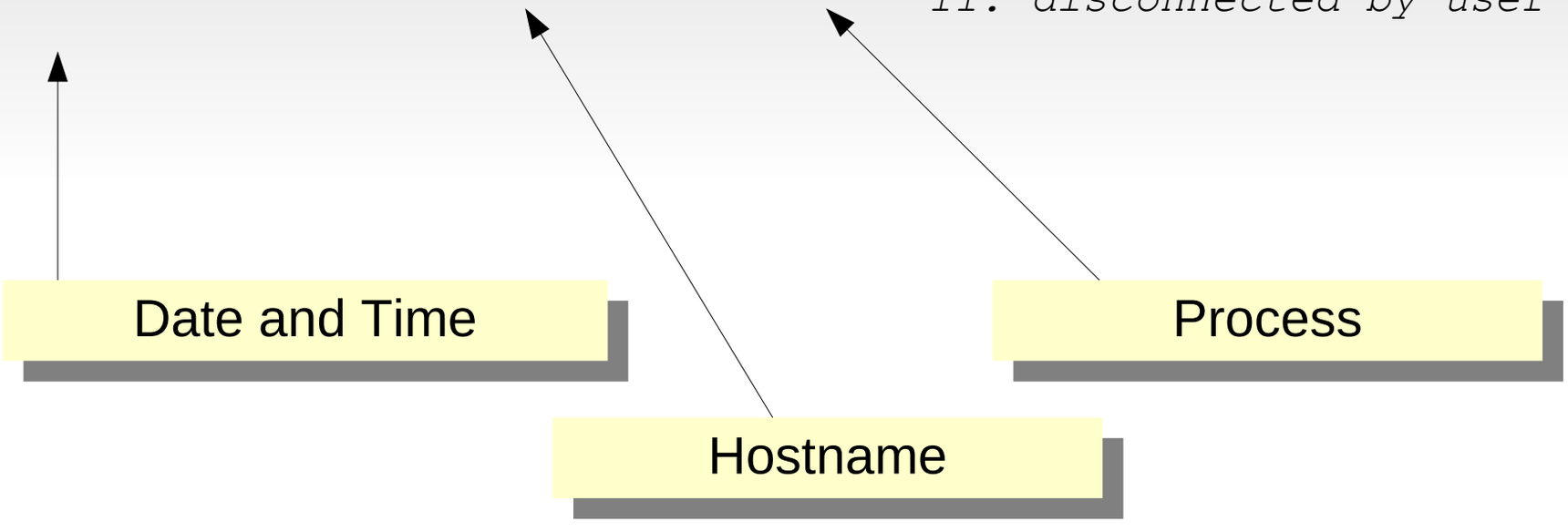
- There are two ways to log events under Linux
  - The program can write the event directly into a log file
  - The program can sent the event to the `syslog` daemon, which then decides if and where to save the message

- syslog was originally developed at UC Berkeley for the `sendmail` daemon
  - Many implementations exist (dsyslog, rsyslog, …)
  - A configuration file in `/etc/` specifies which events are logged and where
  - Logs can be stored in multiple locations (including a remote machine), but they are often found under `/var/log/`

Log Files

- Logs are typically rotated, gzipped, and deleted after a period of time
- Log entries are time-stamped but do not have a fixed structure

```
May 31 23:23:26 crazyivan sshd[2413]: Received disconnect from 15.136.123.14:
                                      11: disconnected by user
```



Log Files

- Traditional logs include:
  - `kern.log` – Kernel-related operations
  - `syslog` – Depends on the configuration, it may get almost everything
  - `auth.log` – Authentication log
  - `messages` – general, non critical, system activity
  - `wtmp` – Login and logout history log (binary format!!)
  - `dmesg` – Boot log
  - `iptables` – firewall log
- Other logs you may want to check
  - Application specific logs (`apache`, `mysql`, `sendmail`, ...)
  - Package manager log (e.g. `dpkg.log`)

Log Forensics

kern.log

- Network cards enter and leave promiscuous mode
- Booted kernel image
- Sleeps and wakeups

auth.log

- Succesfull and failed sudo commands
- Users login
- Password guessing attacks

daemon.log (or syslog)

- Wpa-supplicant, NetworkManager (wireless network connection)
- dhcpclient

Log Forensics

syslog (or kern.log)

- Connected and disconnected usb devices

last -f /var/log/wtmp

- All users login and logout activity (and login source)
- System reboot

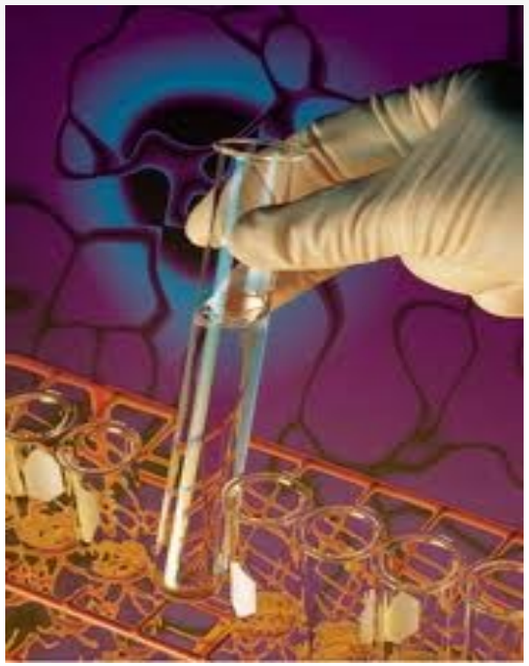
Check the application logs for specific attacks

- e.g., Apache logs for sign of sql injections, or `php` shells

The Microsoft World

Summary

- ✔ OS-independent
- ✔ Linux Artifacts
- Windows Artifacts
  - Registry
  - Recycled Bin
  - Prefetch
  - Shortcut Files
  - Shellbags
  - Thumbs.db
  - Event Logs



Windows Registry

- Hierarchical database (stored in a number of binary files) that contains information and settings about:
  - Users
  - Operating System settings
  - Applications
  - Hardware devices
  - Events
- Data is organized in a number of root Hives
  - Some are permanently stored on disk, some are volatile and they are only populated at runtime
- Inside an hive, information are stored in Cells

Hives

HKEY_CLASSES_ROOT

- Association between files and applications used to open them

HKEY_USERS

- All user profiles

HKEY_CURRENT_USER

- The current logged-in user

HKEY_LOCAL_MACHINE

- System, software, and hardware configuration

HKEY_CURRENT_CONFIG

- The hardware profile used at startup

SAM
SECURITY
SYSTEM
SOFTWARE

Cells

- Registry Key cell
  - Only cell with a timestamp (*LastWrite* time), expressed as number of 100-nanoseconds since midnight 1601
- Value cell
  - Contains a Name, a Data Type, and the Data itself
- Subkey cell
- Security descriptor cell
  - Security information for a key cell

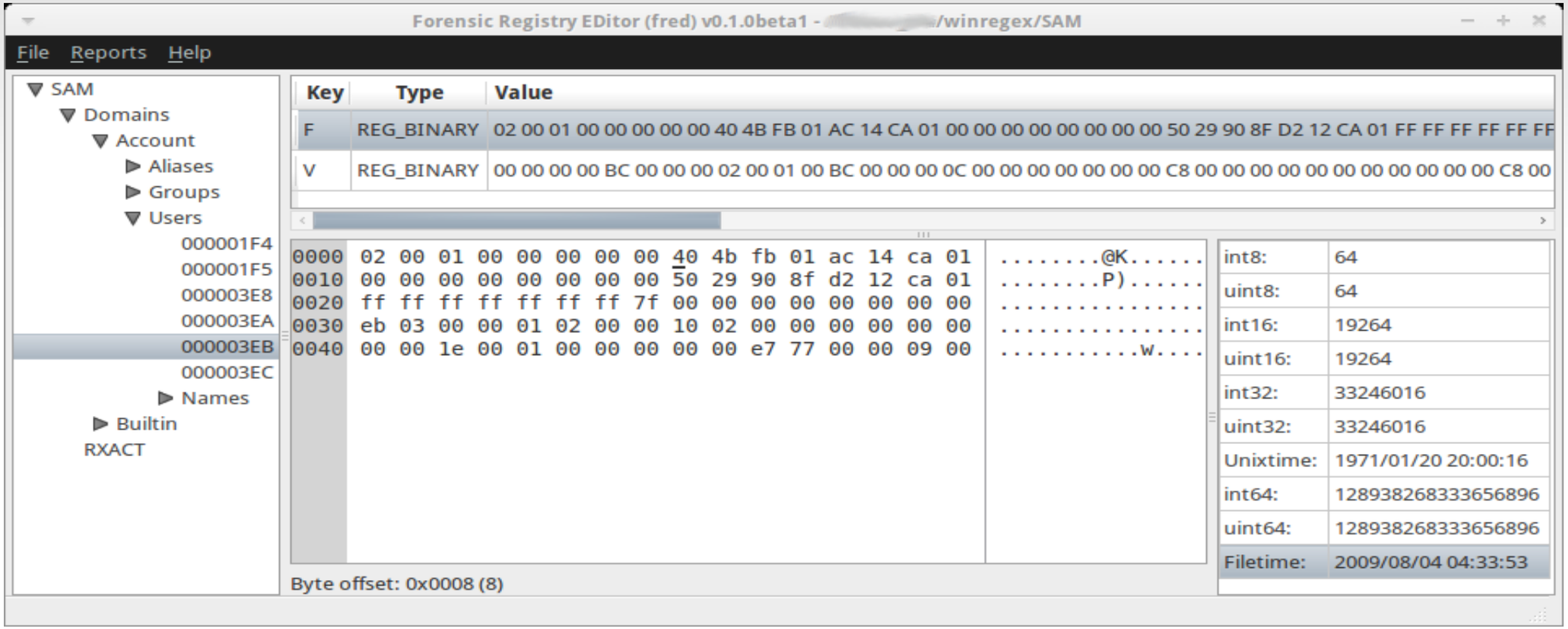
- Note: when a cell is deleted, it is not physically removed from the registry file

Location of Registry Files

- HKEY_LOCAL_MACHINE \SYSTEM `\system32\config\system`
- HKEY_LOCAL_MACHINE \SAM `\system32\config\sam`
- HKEY_LOCAL_MACHINE \SECURITY `\system32\config\security`
- HKEY_LOCAL_MACHINE \SOFTWARE `\system32\config\software`
- HKEY_USERS \UserProfile `\winnt\profiles\username`
- HKEY_USERS.DEFAULT `\system32\config\default`
- HKEY_LOCAL_MACHINE \HARDWARE *volatile hive*

Forensic Registry Editor (Fred)

- Cross-platform registry hive viewer
  - https://www.pinguin.lu/index.php



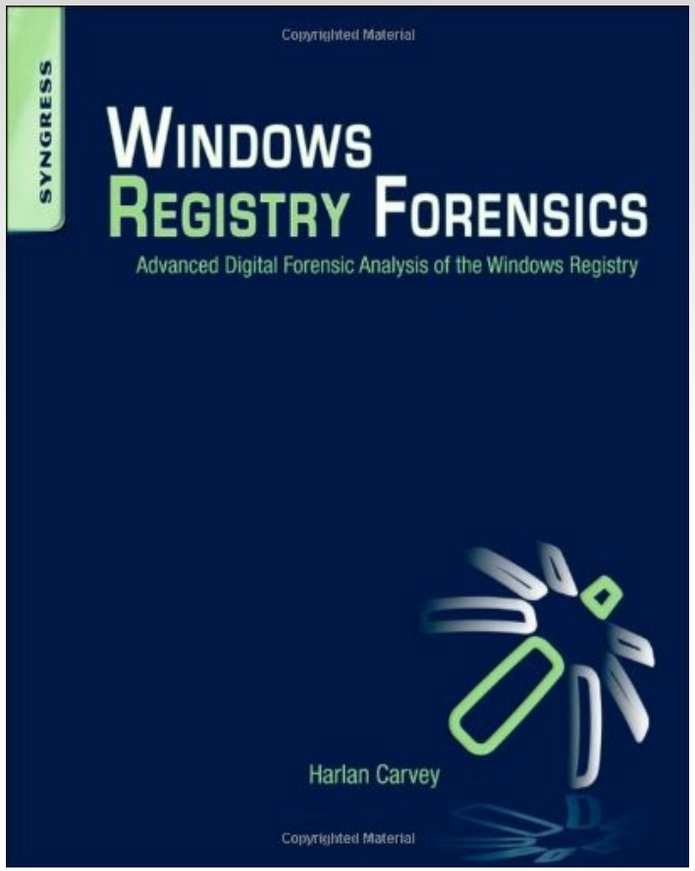
RegRipper

- Forensic registry analysis tool written in Perl
  - It is a data extraction and correlation tool
  - It is *not* a viewer to explore the registry

- RegRipper uses a number of plugins to access Windows registry hives and extract specific keys and values
- Available with both a graphic interface and a command line interface (running on Linux)

Where & What to Look For

- There is a lot of forensically valuable information in the Registry
- Unfortunately...
  - ...most of the information is not documented
  - ...most of the information is application-specific



System Information

- Computer name

```
$ rip.pl -p compname -r ./system
```

- OS version

```
$ rip.pl -p winnt_cv -r ./software
```

- Timezone

```
$ rip.pl -p timezone -r ./system
```

- NTFSDisableLastAccessUpdate

```
$ rip.pl -p disablelastaccess -r ./system
```

Wireless SSID

- Windows maintains a list of the wireless network to which it connected in the past

  `$ rip.pl -p ssid -r ./software`

  `$ rip.pl -p networklist -r ./software` (for Vista)

  - Lists the SSID, the time in which the computer last connected to each of them, and the MAC address of the access point
  - The MAC of the access point can be geolocated by using services like *Skyhook*

Autostart Locations

- Allow applications to be launched without any user interaction
  - Often used by malware to re-load themselves in memory after a reboot
- Many, *many*, **many** available alternatives that belong to several classes:
  - System boot
  - Triggers on user activity
  - Triggers on user login
  - BHO (DLL automatically loaded by Internet Explorer)

Autostart Location

```
$ rip.pl -p services -r ./system
```

- List the installed services and the “start” type

```
$ rip.pl -p svchost -r ./system
```

- svchost.exe provides a way to run services from DLLs.

```
$ rip.pl -p user_run -r ./NTUSER.dat
```

- Apps that are run when the user logs in

```
rip.pl -p bho -r ./software
```

- List all the installed browser helper objects

Autostart Location

```
$ rip.pl -p cmd_shell -r ./software
```

- What happen when a particular type of file is executed

```
$ rip.pl -p appinitdlls -r ./software
```

- DLLs automatically loaded in memory when a GUI application is started

```
$ rip.pl -p notify -r ./software
```

- DLLs that are notified when certain events occur

  (used by a lot of different malware)

```
$ rip.pl -p imagefile -r ./software
```

- Allow the user to specify a debugger to be automatically run when an application starts

Removable Storage Devices

- Whenever an USB external device is connected to the computer, footprints are left in the registry
  - All the unique devices ever connected to the system:

    ```
    $ rip.pl -p usbstor -r ./system
    ```

  - Last time each device was connected:

    ```
    $ rip.pl -p devclass -r ./system
    ```

  - Mount points:

    ```
    $ rip.pl -p mountdev -r ./system
    ```

Users & User Activity

- The Security Accounts Manager (SAM) hive contains information about all the accounts and groups in the system

```
$ rip.pl -p samparse -r ./SAM
```

- In the NTUSER hive, Windows stores (in ROT13!!) all the applications used by the user, with a counter

```
$ rip.pl -p userassist -r ./NTUSER.DAT
  Mon Sep 26 22:56:32 2005 (UTC)
    UEME_RUNPATH:C:\WINDOWS\system32\cmd.exe (1)
  Mon Sep 26 22:49:11 2005 (UTC)
    UEME_RUNPIDL:%csidl2%\Internet Explorer.lnk (14)
```

Timestamp

Counter

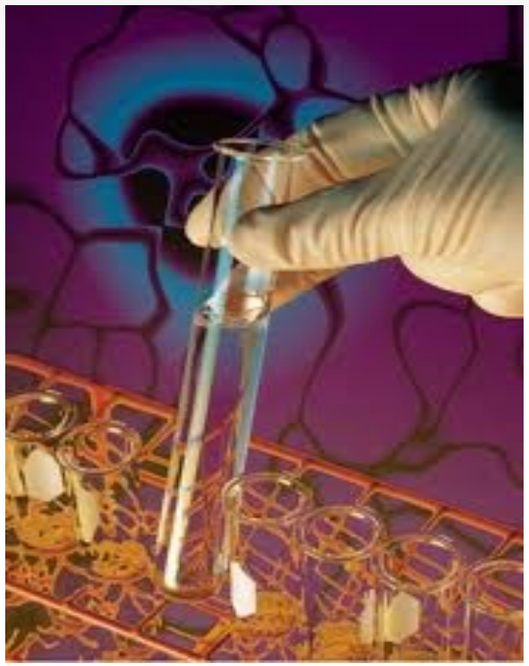
RUNPATH: doubleclick on explorer or through the run box
RUNCPL: control panel applet
RUNPIDL: link (e.g. in START → DOCUMENTS)

User Activity

- Many applications keep in the registry a list of the most recent open documents
  - `$ rip.pl -p runmru -r ./NTUSER.DAT`
    - Most recent command typed in the RUN window
  - `$ rip.pl -p recentdocs -r ./NTUSER.DAT`
    - Most recent documents, grouped by data type
  - `$ rip.pl -p comdlg32 -r ./NTUSER.DAT`
    - Shows the files that have been OPEN or SAVED through the Open/Save dialog box
  - `$ rip.pl -p typedurls -r ./NTUSER.DAT`
    - Most recent URLs typed in the Internet Explorer address bar
  - `$ rip.pl -p acmru -r ./NTUSER.DAT`
    - Last queries used in the file and folder search tool

Summary

- ✔ OS-independent
- ✔ Linux Artifacts
- ✔ Windows Artifacts
  - ✔ Registry
  - Recycled Bin
  - Prefetch
  - Shortcut Files
  - Shellbags
  - Thumbs.db
  - Event Logs



Recycled Bin

- A storage location used to keep a copy of the deleted files, so the user can easily recover them
- From Windows Vista..
  - `\$Recycle.Bin\%UserId%\`
  - `$R<random>.extension` → the original file
  - `$I<same_random>` → file metadata (original path, size, and deletion time)
  - `$I` are binary files of 543 bytes
    (bytes 8-15: filesize, bytes 16-23: timestamp, bytes 24-543: original name)

Prefetch

- Introduced in Windows XP to increase the system performance
  - Boot prefetching
    - The cache manager monitors all page faults that occur in the first 2 minute after boot (or 1 minute after services started)
  - Application prefetching
    - The cache manager monitors the first 10 seconds after a process is started
- The processed data is stored in `.pf` file in the directory `Windows\Prefetch`
  - Limited to 128 files
  - Disabled by default on fast SSD drives

Prefetch

- `pf` files contain interesting forensic information:
  - Unicode list of DLLs used by the executable
  - Number of time the application has been launched
  - Timestamp of last execution
  - The creation date of the `pf` file is the date in which the application was run for the first time
- The file formats have been reverse engineered
  - Two Perl scripts were released by Harlan Carvey to parse them
  - Or you can write you own code
- Vista introduced SuperFetch, which works with the memory manager to optimize the memory content for a given user for a given time of day

Shortcut Files (lnk)

- Metadata stored in a `.lnk` file
  - Path to the target document + timestamps (when it was last opened) + info about the volume that contains the document (or the network share) + file attributes (including size)
- Created automatically, when the user access a file to populate the “recent items” (e.g., jumplist in Windows 7)
  - WinXP location:

    ```
    \Documents and Settings\UserName\Recent
    \Documents and Settings\UserName\Application Data\Microsoft\Office\Recent
    ```

  - Windows 7 (and later) location:

    ```
    \Users\UserName\AppData\Roaming\Microsoft\Windows\Recent Items
    \Users\UserName\AppData\Roaming\Microsoft\Office\Recent Items
    ```

Shellbags

- Registry keys used to store the user preferences to display folders in Explorer
  - Folder name, full path, how items were listed, size,...
  - Timestamps: first accessed & last updated
  - A shellbag exists for each folder ever opened in Explorer !!

- `LNK` files exist only for opened files, while shellbags also exist if the user navigated <u>through</u> a directory
  - Including folders in external medias or encrypted drives

Thumbs.db

- Hidden file (one per directory) that contains the thumbnails of the pictures
  - The images are stored in OLE format (the same used by MS Office)
  - Thumbnails are created also for other file formats (e.g., PDF, DOC, ...)
  - When a file is deleted from the filesystem, the related thumbnail and associated metada remain in the Thumbs.db file (!!)
- On Linux, the content of the `Thumbs.db` file can be extracted using the `vinetto` python script
- In windows Vista / 7 / 8, all the db have been centralized, and stored under:

  `Users\%username%\AppData\Local\Microsoft\Windows\Explorer`

File metadata:

Directory:	/Documents and Settings/jfbeckers/Desktop/my_shoots
Filename:	Thumbs.db
Modification:	Fri May 12 08:10:00 2006
File size:	157696
MD5 digest:	975a0179aa0461818dbb7b5c09afa606

Root Entry modify timestamp : Fri May 12 08:09:59 2006

1


0001


0002


0003


0004


0005

```
0001 -- Wed Mar 22 16:48:32 2006 -- ssa50330.jpg
0002 -- Sun Dec 18 21:28:38 2005 -- 100_2084.JPG
0003 -- Sun Dec 18 21:27:12 2005 -- 100_0866.jpg
0004 -- Sun Dec 18 21:27:12 2005 -- 100_0883.jpg
0005 -- Sun Dec 18 21:27:14 2005 -- 100_0889.jpg
```

2


0006


0007


0008


0009


0010

Event Log

- Roughly equivalent to the Linux syslog
  - In binary format
  - Each event is represented by an ID. The real message is stored in an external DLL to allow for internationalization
  - You can ask Google for the `event_id` to find an English description
- The setup and location of the logs is stored in the registry

```
$ rip.pl -p eventlog -r system
```

Event Log Analysis

- Suite of perl scripts by Harlan Carvey

```
$ evtstats.pl AppEvent.Evt
 Max    Size of the Event Log file            = 65536 bytes
 Actual Size of the Event Log file            = 65536 bytes  Total number of
event records (header info)  = 200
 Total number of event records (actual count) = 206
 Total number of event records (rec_nums)     = 206
 Total number of event records (sources)      = 206
 Total number of event records (types)        = 206
 Total number of event records (IDs)          = 206
```

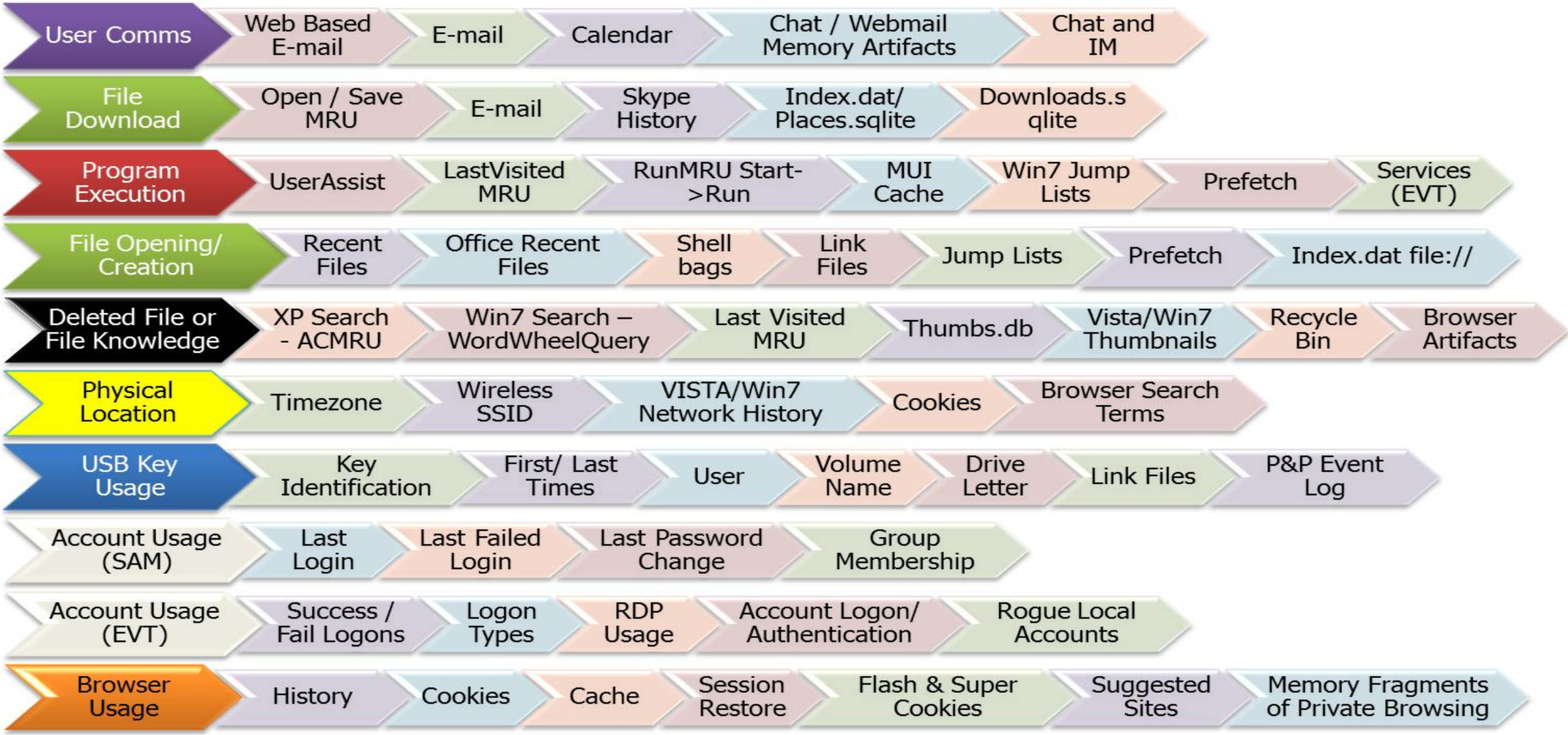
```
$ evtrpt.pl SecEvent.Evt
```

- Prints the distribution of event ids

```
$ lsevt.pl SecEvent.Evt
```

- Lists each event one by one

General Event Descriptions	General Event IDs
Account and Group Activities	4624, 4625, 4648, 4728, 4732, 4634, 4735,4740, 4756
Application Crashes and Hangs	1000 and 1002
Windows Error Reporting	1001
Blue Screen of Death (BSOD)	1001
Windows Defender Errors	1005, 1006, 1008, 1010, 2001, 2003, 2004, 3002, 5008
Windows Integrity Errors	3001, 3002, 3003, 3004, 3010 and 3023
EMET Crash Logs	1 and 2
Windows Firewall Logs	2004, 2005, 2006, 2009, 2033
MSI Packages Installed	1022 and 1033
Windows Update Installed	2 and 19
Windows Service Manager Errors	7022, 7023, 7024, 7026, 7031, 7032, 7034
Group Policy Errors	1125, 1127, 1129
AppLocker and SRP Logs	865, 866, 867, 868, 882, 8003, 8004, 8006, 8007
Windows Update Errors	20, 24, 25, 31, 34, 35
Hotpatching Error	1009
Kernel Driver and Kernel Driver Signing Errors	5038, 6281, 219
Log Clearing	104 and 1102
Kernel Filter Driver	6
Windows Service Installed	7045
Program Inventory	800, 903, 904, 905, 906, 907, 908
Wireless Activities	8000, 8001, 8002, 8003, 8011, 10000, 10001, 11000, 11001, 11002, 11004, 11005, 11006, 11010, 12011, 12012, 12013
USB Activities	43, 400, 410
Printing Activities	307



*from SANS forensics

SANS Windows Artifacts Poster