/Rooted® 2014

Mac OS X Forensics

En profundidad

V

getuid

- Joaquín Moreno (bastionado@gmail.com, @moxilo)
- Pentester 4 work, Forensics 4 fun ☺
- Coleccionista de cromos.
- MSc Information Security en RHUL.
  - Parte de mi tesina (termina en septiembre)
  - Tutor: Lorenzo Cavallaro
  - https://code.google.com/p/mac-osx-forensics/
- Colaborador del proyecto Plaso.

PLASO: Plaso Langar Að Safna Öllu

- Tolo lo que veremos está siendo implementado en Plaso.
- Antiguo Log2Timeline (NO es una mejora).
- Forensics framework (http://plaso.kiddaland.net/)
- Capacidad de trabajar sobre imágenes de disco (raw, encase, qemu, vhdi), particiones, puntos de montaje o evidencias individuales.
- File Vault2 y Bitlocker librerías disponibles (no en core).
- Extracción automatizada de localtime, usuarios y otra información preliminar.

PLASO: Plaso Langar Að Safna Öllu

- Línea de tiempo en NTFS, FAT, EXT2/3/4, HFS+, etc.
- Evidencias de SSOO en Windows (EVT/X, Prefetch, lnk, Registro, VSS, Jobs, etc), Linux (UTMP, Syslog, Selinux, etc), Mac OS X (ASL, BSM, Keychain, etc) y Android.
- Evidencia de aplicaciones como Chrome, Safari, IE, Firefox, Skype, Java IDX, Google Drive, servicios, etc.
- Fácil de usar ☺

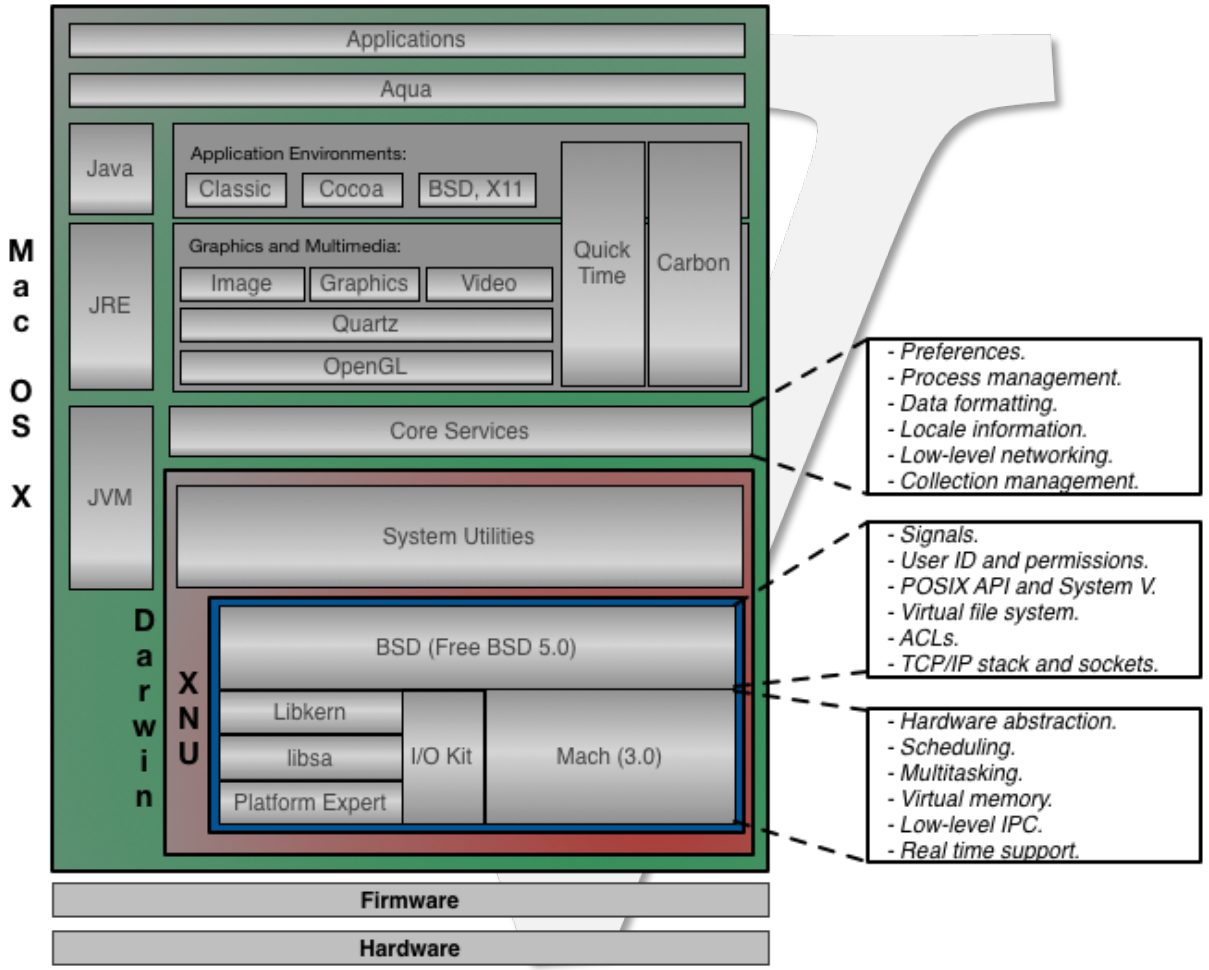
  $ log2timeline.py –partition_map imagen_disco

  $ log2timeline.py –partition 2 resultado imagen_disco
- Consola (tipo SQL): psort.py, GUI: 4n6time y Kibana.

¿Qué vamos a ver?

- Evidencias en tres niveles:
  1. Sistema de ficheros (HFS+).
  2. **Evidencias del sistema operativo.**
  3. Evidencias propias de la aplicación.
- Muy poca documentación del segundo nivel:
  - ¿Hay información interesante?
  - ¿Cómo se extrae?
  - No vale usar programas de Apple...

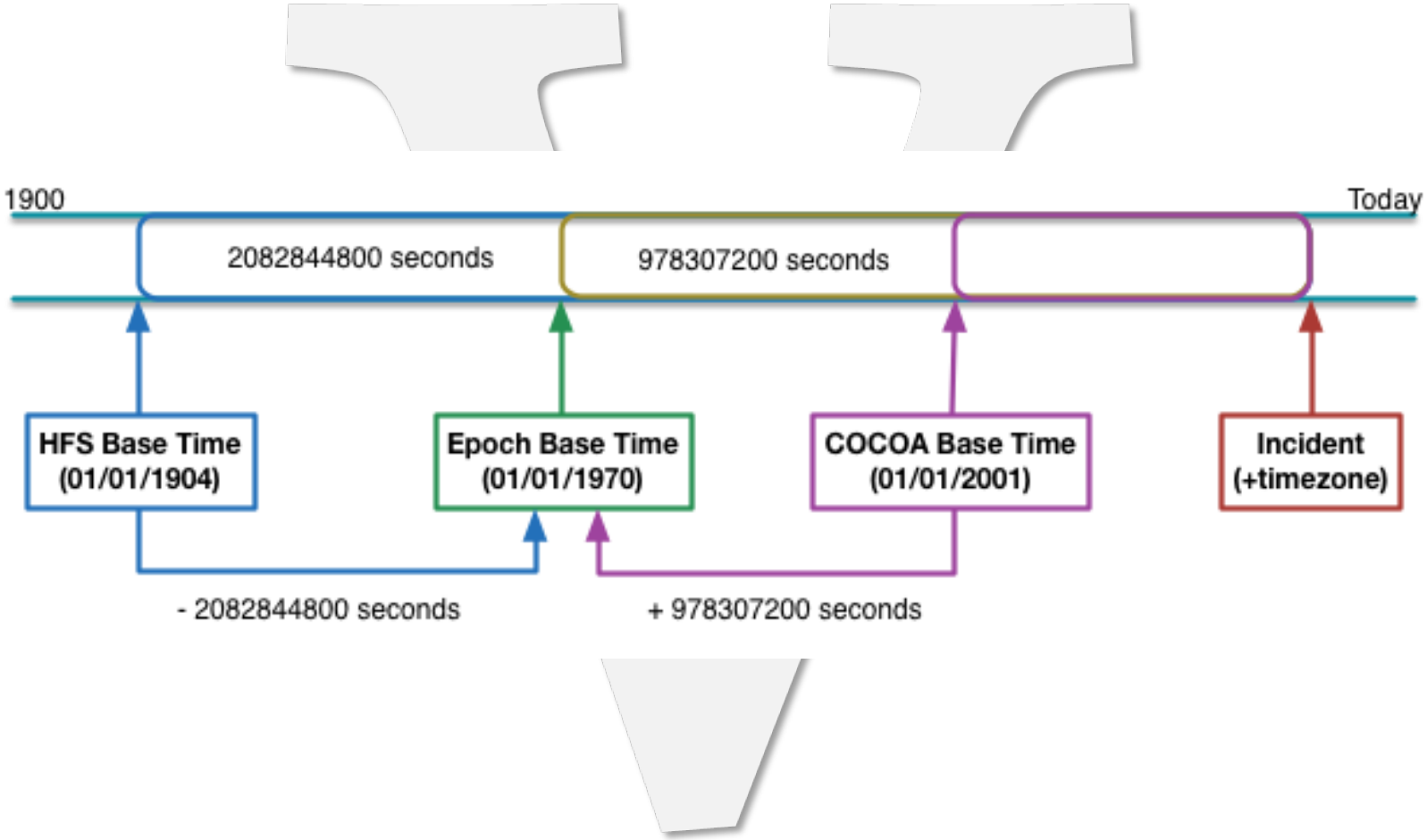
Mac OS X


Applications
Aqua
Java
Application Environments:
Classic
Cocoa
BSD, X11
JRE
Graphics and Multimedia:
Image
Graphics
Video
Quartz
OpenGL
Quick Time
Carbon
Mac OS X
Core Services
JVM
System Utilities
Darwin
XNU
BSD (Free BSD 5.0)
Libkern
libsa
Platform Expert
I/O Kit
Mach (3.0)
Firmware
Hardware
- Preferences.
- Process management.
- Data formatting.
- Locale information.
- Low-level networking.
- Collection management.
- Signals.
- User ID and permissions.
- POSIX API and System V.
- Virtual file system.
- ACLs.
- TCP/IP stack and sockets.
- Hardware abstraction.
- Scheduling.
- Multitasking.
- Virtual memory.
- Low-level IPC.
- Real time support.

De todo un poco... mezcladito...

- Código fuente no siempre disponible.
- Evidencias binarias en LE y otras en BE.
- Texto en UTF-8, ASCII (7bits) y UTF-16.
- Cuatro diferentes timestamps:
  - HFS: 1-1-1904 UTC
  - Epoch: 1-1-1970 UTC
  - Cocoa: 1-1-2001 UTC
  - Texto: human readable, syslog, propio.

Timestamp


1900
Today
2082844800 seconds
978307200 seconds
HFS Base Time
(01/01/1904)
Epoch Base Time
(01/01/1970)
COCOA Base Time
(01/01/2001)
Incident
(+timezone)
- 2082844800 seconds
+ 978307200 seconds

Índice:

- Adquisición de evidencias
- Apple System Log (ASL)
- UTMPX
- Basic Security Module (BSM)
- Keychain
- Document Version
- CUPS IPP
- Lista de propiedades (Plist)

Adquisición de evidencias

- Desmontar y copiar… No siempre se puede.
- Modo solo lectura: “Cmd + s” (¿seguro?)
  - Mac nativo solo soporta escribir en HFS.
  - dd bs=512 conv=noerror,sync if=/dev/disk0 of=/mnt/mac_osx.dd
- Linux live CD: “c” (mejor)
  - dd bs=512 conv=noerror,sync if=/dev/sda of=/mnt/image.dd
- Si SSD con TRIM: NO usar DC3DD.
- Montar partición (mmls para conocer X):
  - mount -t hfsplus -o loop,offset=$((X*512)),ro,noexec,umask=0222 image.dd /mnt/mac
  - Si FAIL! ☹ Entonces Kpartx!

Kpartx!

- # kpartx –av image.dd

```
root@plaso:~/rooted# kpartx -av /mnt/hgfs/moxilo/Rooted/malware.dd
add map loop0p1 (252:2): 0 409600 linear /dev/loop0 40
add map loop0p2 (252:3): 0 82206864 linear /dev/loop0 409640
add map loop0p3 (252:4): 0 1269536 linear /dev/loop0 82616504
root@plaso:~/rooted# OPTION='loop,noexec,ro,umask=0222'
root@plaso:~/rooted# mount -t hfsplus -o $OPTION /dev/mapper/loop0p2 mount/
root@plaso:~/rooted# ls -l mount/mach_kernel
-rwxr-xr-x 1 root root 8393256 Sep 20 06:22 mount/mach_kernel
root@plaso:~/rooted# umount mount/
root@plaso:~/rooted# kpartx -dv /mnt/hgfs/moxilo/Rooted/malware.dd
del devmap : loop0p3
del devmap : loop0p2
del devmap : loop0p1
loop deleted : /dev/loop0
root@plaso:~/rooted#
```

File Vault 2: libfvde!

- https://code.google.com/p/libfvde/
- EncryptedRoot.plist.wipekey se extrae de la partición Recovery HD usando TSK (fls + icat)
- # fvdemount -X allow_root -e EncryptedRoot.plist.wipekey -p password -o $((512*offset)) filevault2.dd mount/
- # dd bs=512 conv=noerror,sync if=mount/fvde1 of=root_partition
- # mount –t hfsplus –o loop,ro,... mount/fvde1 mount2/
- Próximamente en Plaso.

LibFVDE:

```
root@plaso:/images# mmls filevault2_10.9.dd | grep "Recovery\|Mac"
05:  01      0000409640   0082616503   0082206864   Macintosh HD
06:  02      0082616504   0083886039   0001269536   Recovery HD
root@plaso:/images# fls -r -o 82616504 filevault2_10.9.dd | grep EncryptedRoot
+++++ r/r 180:  EncryptedRoot.plist.wipekey
root@plaso:/images# icat -o 82616504 filevault2_10.9.dd 180 >> ER
root@plaso:/images# xxd -l 10 ER
0000000: 3343 8e0c 907a 2db3 ccb4                 3C...z-...
root@plaso:/images# OF=$((512*409640))
root@plaso:/images# fvdemount -e ER -p abcd -o $OF filevault2_10.9.dd mount/
fvdemount 20130305

root@plaso:/images# mkdir mount2
root@plaso:/images# OPT='ro,noexec,loop,umask=0222'
root@plaso:/images# mount -t hfsplus -o $OPT mount/fvde1 mount2/
root@plaso:/images# ls -l mount2/mach_kernel
-rwxr-xr-x 1 root root 8393256 Sep 20 06:22 mount2/mach_kernel
root@plaso:/images#
```

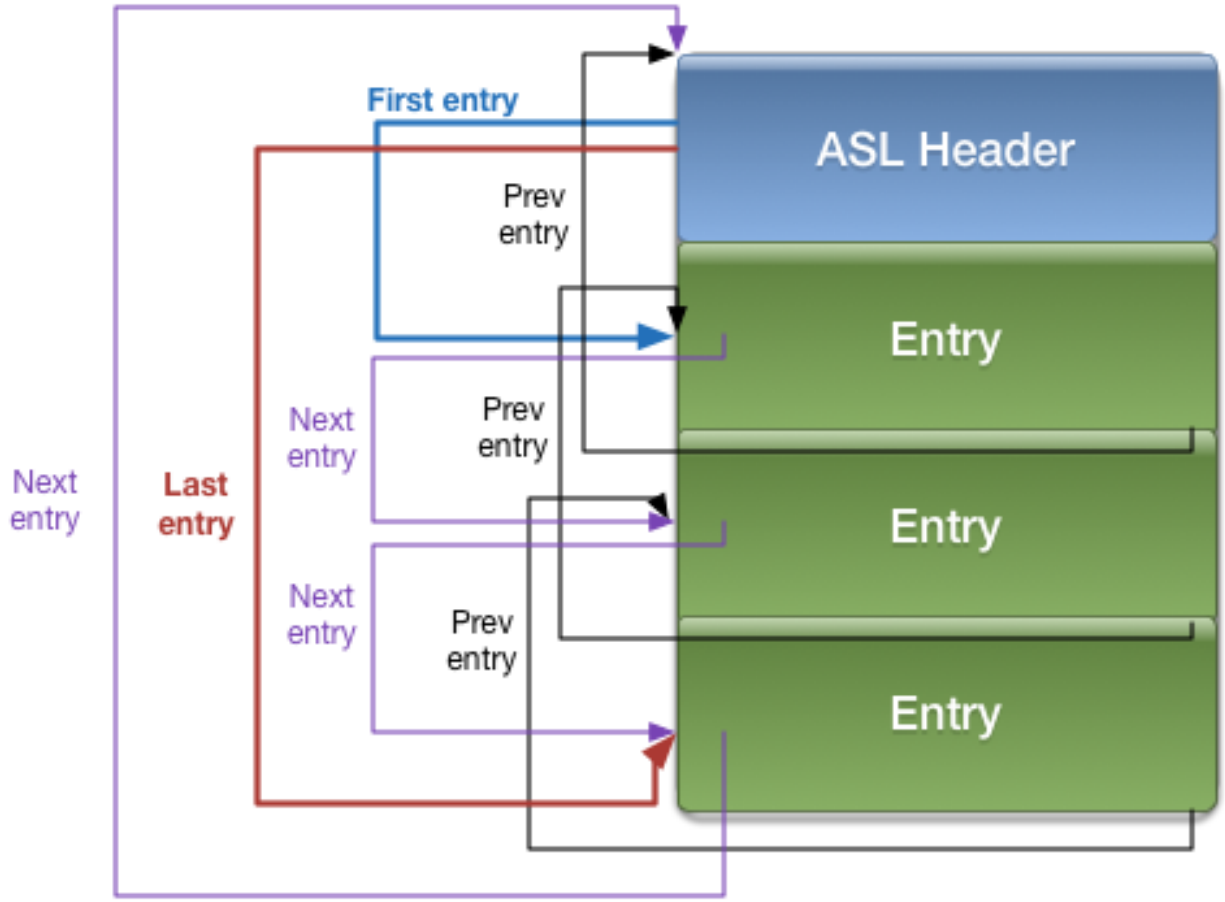
Apple System Log

- Inicializado por: com.apple.syslogd.plist
- Binario: /usr/sbin/syslogd
- Configurado: /etc/asl.conf y /etc/asl/*
- Rotado: aslmanager
- En 10.5 de texto plano a **formato binario**.
- Principalmente en: /private/var/log/asl/
- Formato fichero: YYYY.MM.DD.[UID].[GID].asl

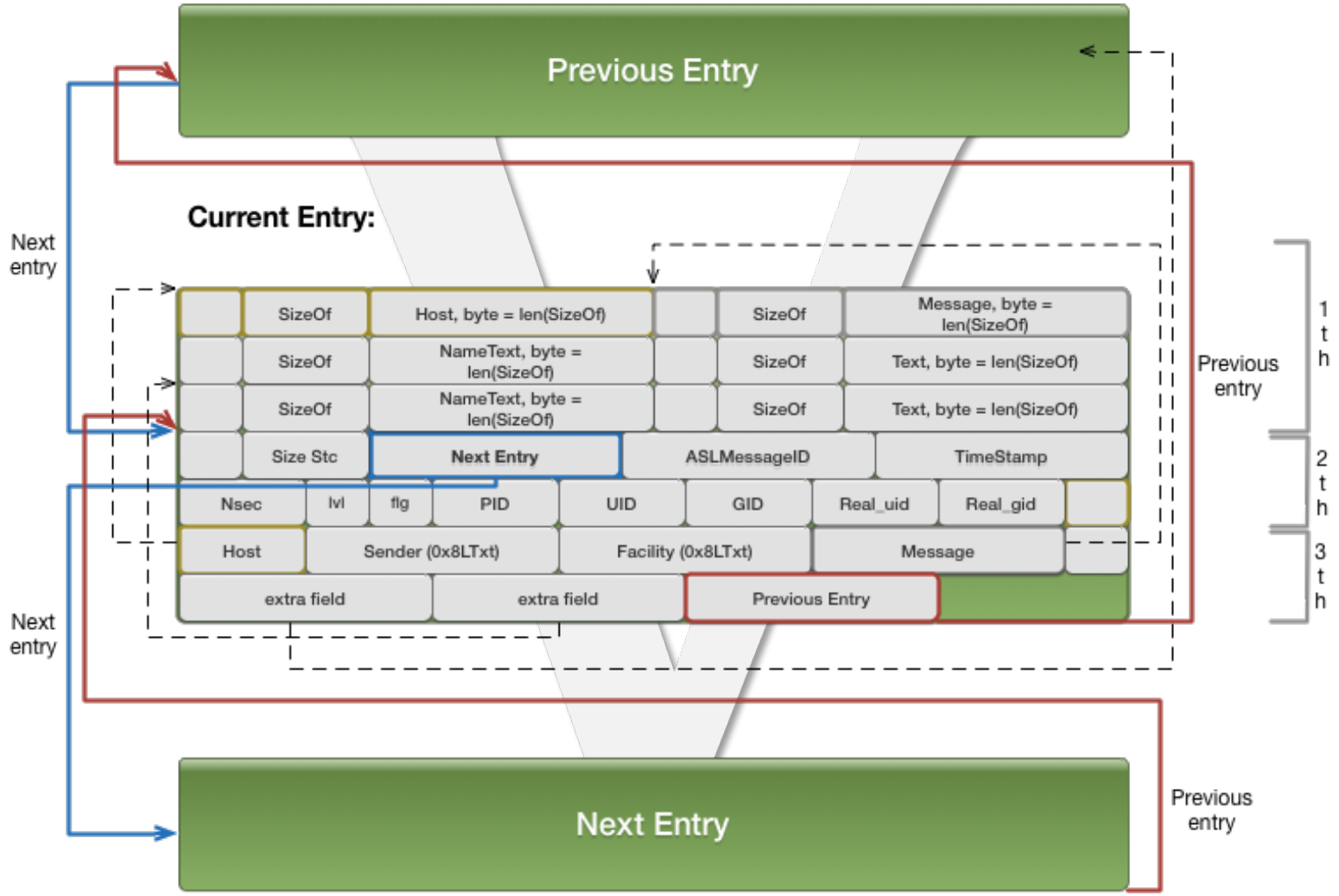
ASL Formato binario

- Epoch timestamp con microsegundos.
- Estructura en Big Endian.
- Cabecera: puntero a la primera y última entrada.
- Doble lista enlazada de entradas.
- Puntero: 8 bytes con la posición absoluta dentro del fichero (número de bytes).
- Apple: syslog -f file.asl -T utc -F raw

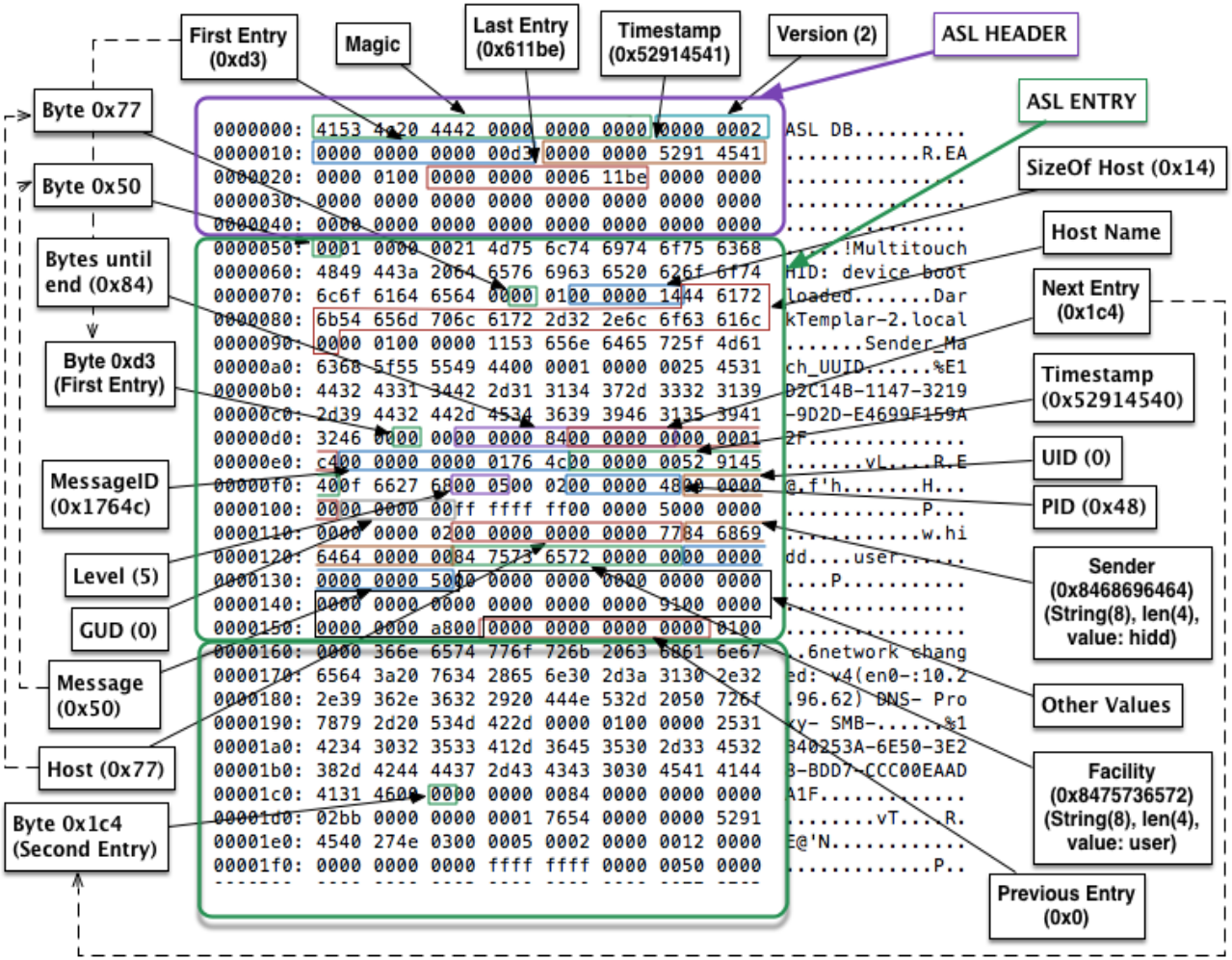
ASL formato binario



Entrada ASL


Previous Entry
Current Entry:
Next entry
SizeOf
Host, byte = len(SizeOf)
SizeOf
Message, byte = len(SizeOf)
SizeOf
NameText, byte = len(SizeOf)
SizeOf
Text, byte = len(SizeOf)
SizeOf
NameText, byte = len(SizeOf)
SizeOf
Text, byte = len(SizeOf)
Size Stc
Next Entry
ASLMessageID
TimeStamp
Nsec
lvl
flg
PID
UID
GID
Real_uid
Real_gid
Host
Sender (0x8LTxt)
Facility (0x8LTxt)
Message
extra field
extra field
Previous Entry
Previous entry
1 t h
2 t h
3 t h
Next entry
Next Entry
Previous entry

ASL binario de cerca...



Evidencias

- Notificación a nivel de aplicación.
- Acceso al sistema (10.5): UTMP, WTMP, LAST.
- Errores y comportamientos anómalos.
- Conexiones a servicios.
- USB montados: USBMSC + 0xFabricante + 0xModelo
  - http://www.linux-usb.org/usb.ids
- Y más, junto al sistema de fichero, ¡ASL binario es de lo más importante!

ASL BSD plaintext

- Syslog tradicional:

  Month Day HH:MM:SS Host Sender [PID]: Message

- ¿Año? Usamos el Crtime del fichero... ☹
- Mensaje puede ser multilínea.
- “Repeated line X time”: primera y última vez.
- La mayoría de la información valiosa de ASL es almacenada en formato binario.

Los logs más destacados

- BSD Plaintext (hay muchos más...):
  - /private/var/log/authd.log
  - /private/var/log/system.log
  - /private/var/log/install.log
- Log del firewall:
  - /private/var/log/appfirewall.log
- Securityd (muy granular):
  - /var/log/module/com.apple.securityd/security.log.YYYYMMDDTHHMMSSZ

No solo ASL... ¡Hay más!

- **Newsyslog daemon**:
  - Usado para rotar logs (espacio ocupado).
  - Conf: /etc/newsyslog.conf
  - Logs:
    - /Library/Logs/* (No relevante).
    - /private/var/log/wifi.log (Histórico WiFi)
- **Human file format**:
  - Timestamp + mensaje.
  - Cada boot time, rotado y nuevo fichero.

¿Soportado en Plaso?

- Próxima release ☺ :
  - ASL Binario: asl.py
  - Syslog con Pyparsing: syslog.py (CL)
  - Syslog multilínea (BSD): mac_syslog.py (CL)
  - Firewall: mac_appfirewall.py
  - Securityd: mac_securityd.py
  - Wifi.log: mac_wifi.py
  - Human readable: mac_human_syslog.py (CL)

UTMPX

- UTMPX existe por compatibilidad.
- Una entrada UTMPX por cada sesión al sistema.
- Epoch timestamp con microsegundos de cuando se accedió al sistema.
- Estructura en Little Endian (eq to Linux).
- Donde: /private/var/run/utmpx
- Plaso: utmpx.py

UTMPX

- Se crea nuevo cuando arranca el sistema.
- ¡Cuando el sistema se suspende también cuenta!
- En boot time se crea el fichero nuevo con dos entradas UTMPX:
  - User utmpx-1.0 + Status SIGNATURE (0x0A)
  - Status BOOT_TIME (0x02)
- Usuario: si status 0x7 (activa), 0x8(finalizada).

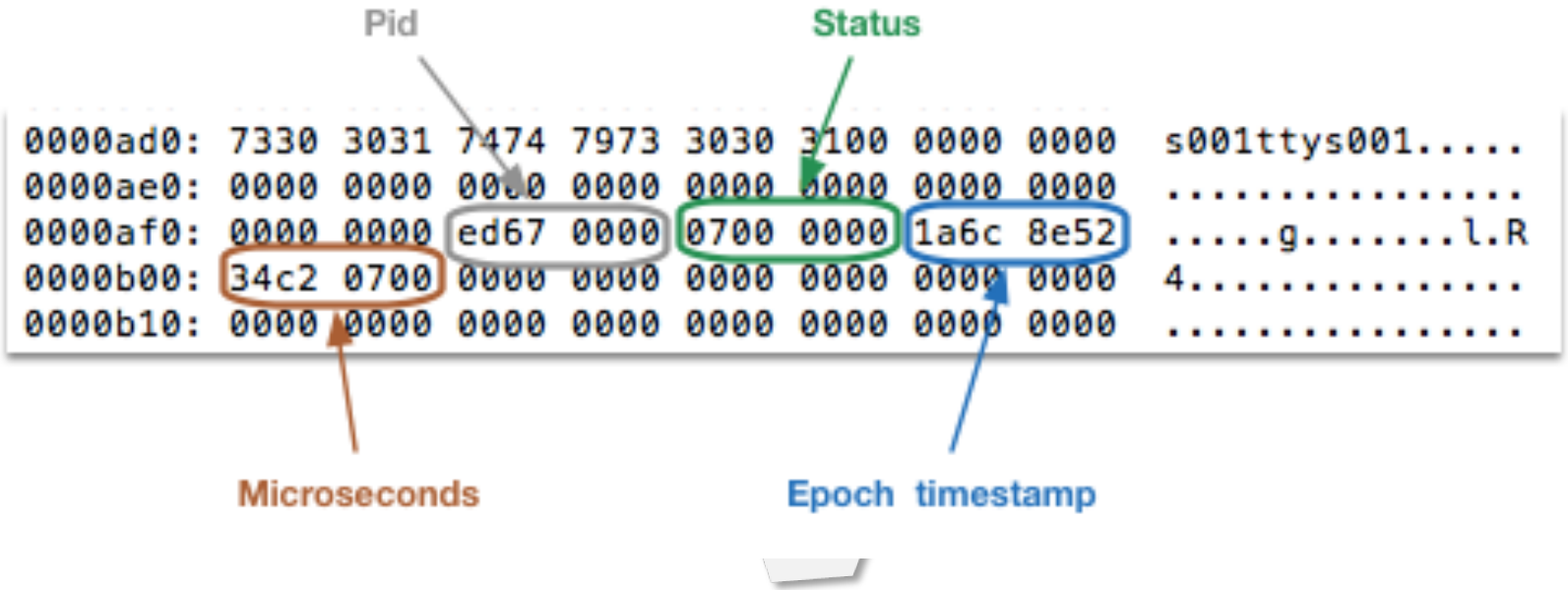
Estructura entrada UTMPX

Order	Name	Length
1th	User name	256 bytes
2th	ID size	4 bytes
3th	Terminal name	32 bytes
4th	Process ID	4 bytes
5th	Session status	*4 bytes
6th	Epoch timestamp	4 bytes
7th	Microsecond	4 bytes
8th	Host name	256 bytes
9th	Padding for future fields	64 bytes

Table: 6.3: UTMPX structure.

¿Status es un short o integer?

- Según la documentación el campo status es un short (2 bytes) sin embargo hay espacio para un integer (4 bytes):



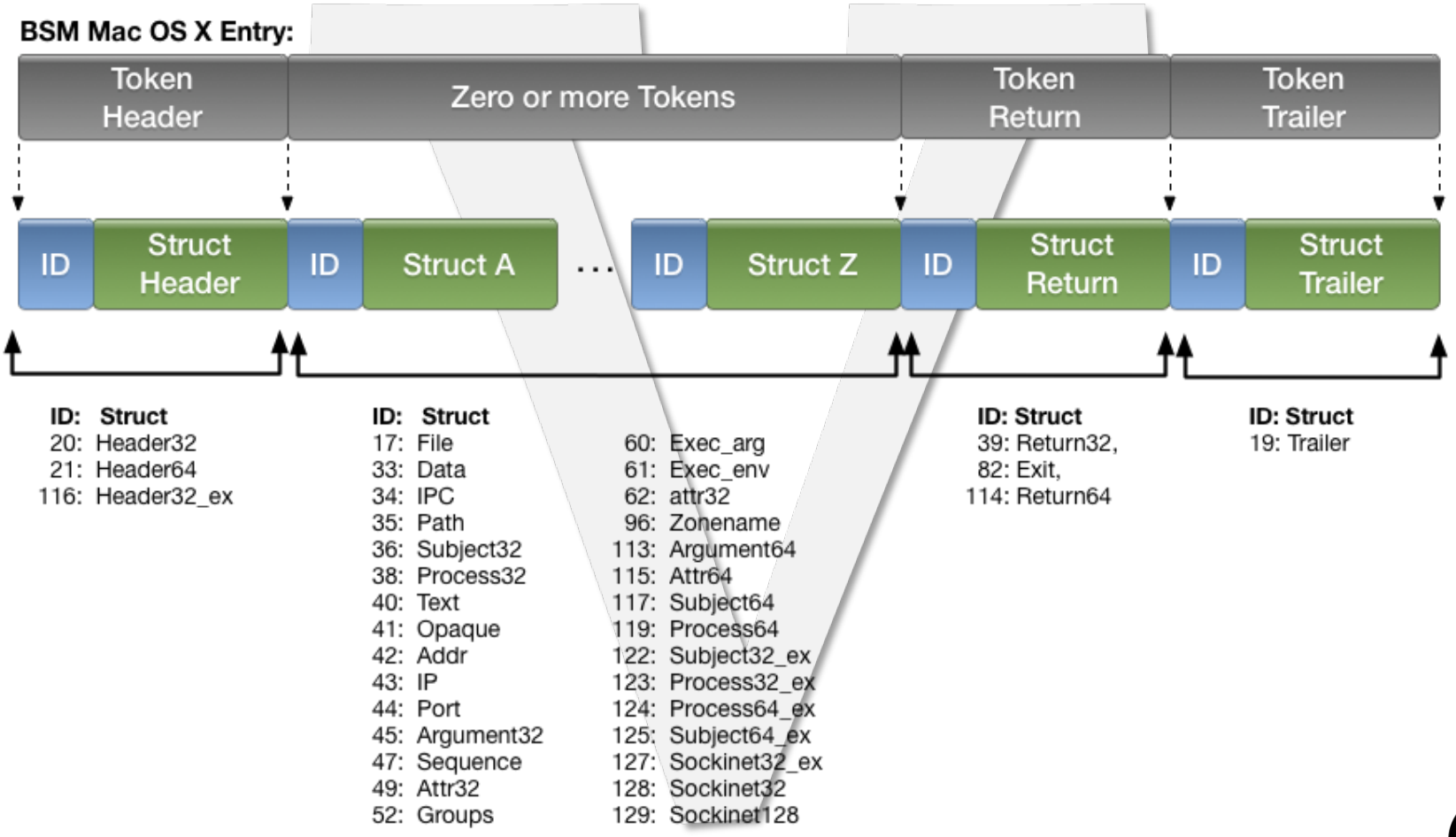
Basic Security Module (BSM)

- Auditoría a nivel del núcleo.
- Creado para Solaris (Nivel C2 en TCSEC).
- Apple crea su propia versión (OpenBSM 1.1):
  - Implementado por McAfee, liberado con licencia BSD (FreeBSD 6.2) y mantenido por TrustedBSD.
- Donde: /private/var/audit/starttime.endtime
- Crucial para análisis de malware.
- Apple: praudit -e bsm_file | Plaso: bsm.py

Basic Security Module (BSM)

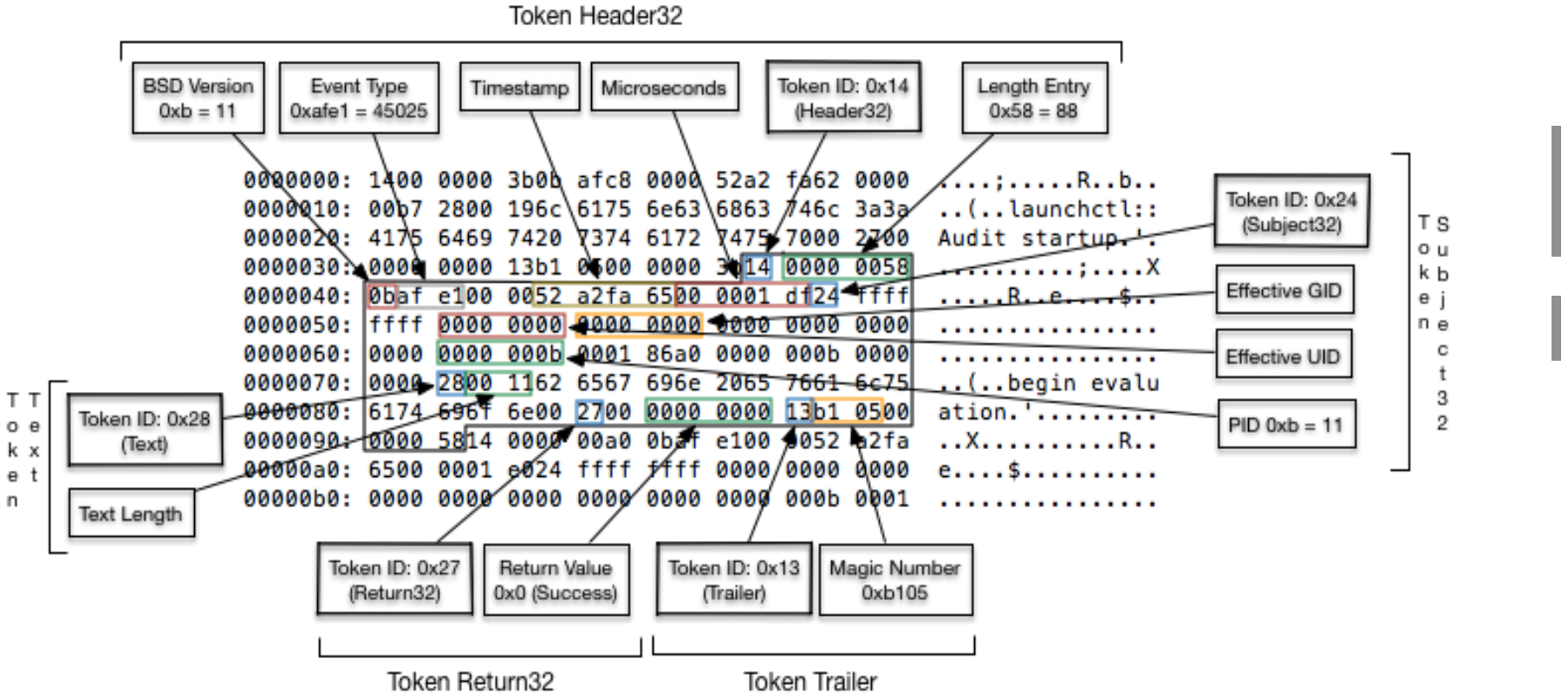
- Cada fichero formado por una o más entradas.
- Una entrada es uno o más tokens.
- Epoch timestamp y formato big endian.
- Obligatorio token header:
  - Tamaño de entrada.
  - Timestamp.
  - ID tipo de evento: /etc/security/audit_event
- Token Return y token Trailer casi siempre.
- 40 Tokens (estructuras binarias) diferentes, 4fun...

Basic Security Module (BSM)



BSM File

BSM Mac OS X Entry:

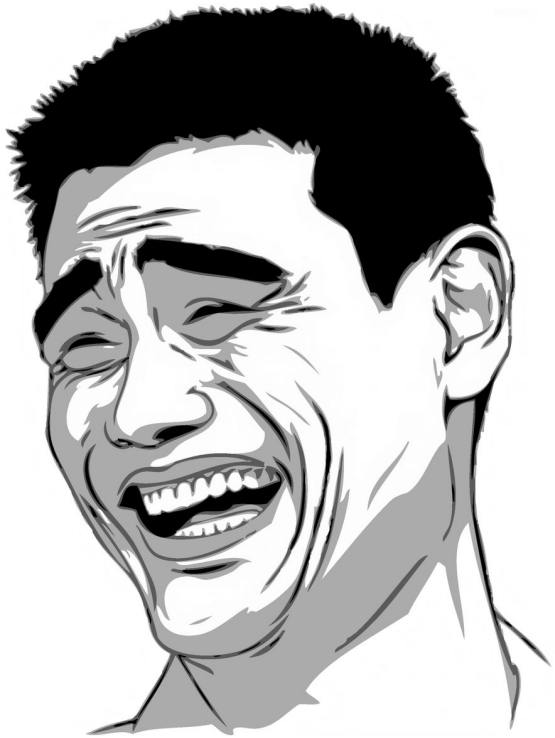


Keychain

- Base de datos binaria que contiene las contraseñas de todo, incluida la mayoría de las aplicaciones, menos los usuarios del sistema.
- Solo las notas secretas y las contraseñas estás cifradas. El resto: URLs, nombre del programa, nombre WiFi, nombre del usuario, protocolos, cuando se guardo por primera vez la contraseña, si se ha cambiado desde entonces y otros valores están en...

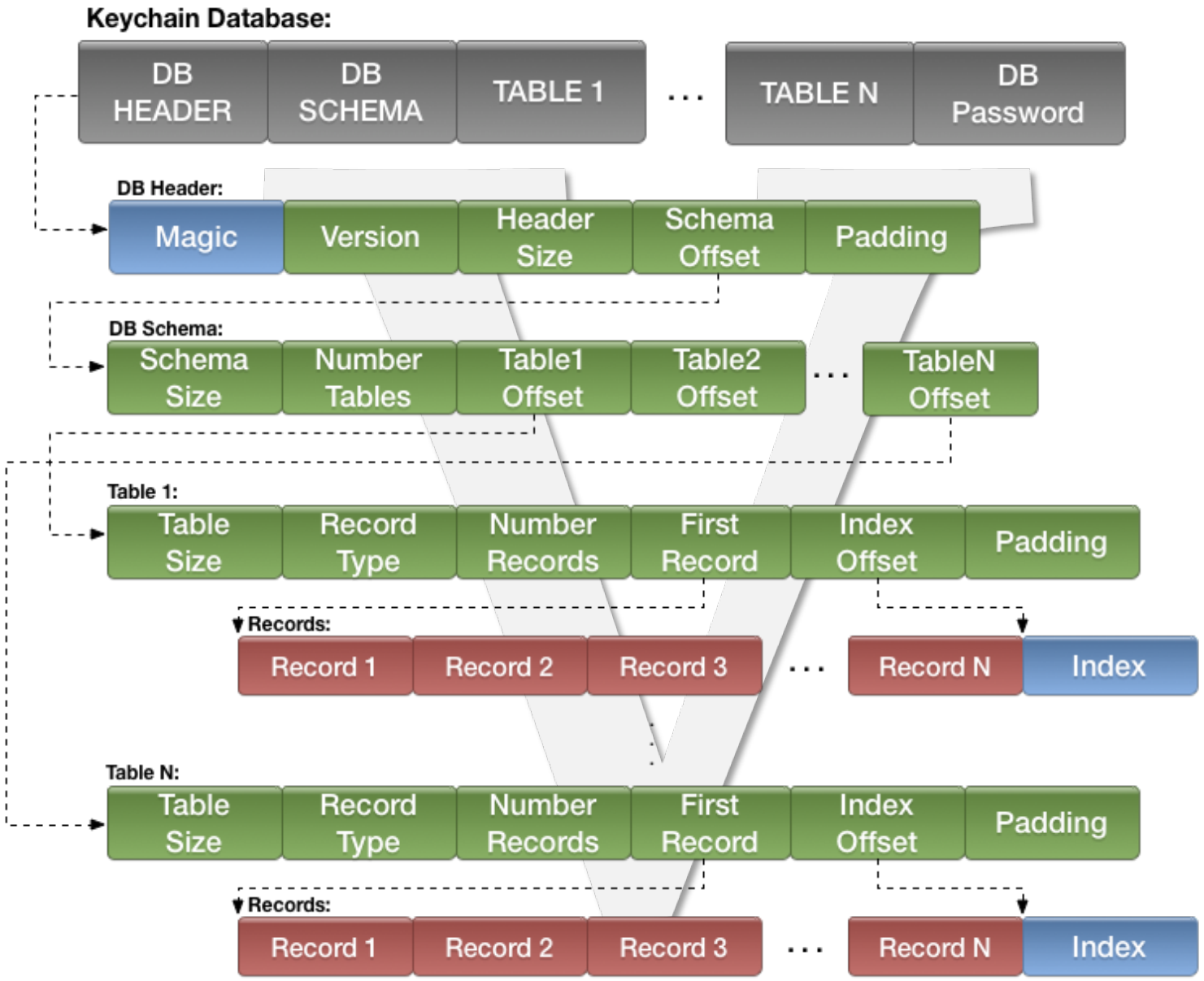
Keychain

- ¡Texto plano!
- Sólo hay que saber leer la info.

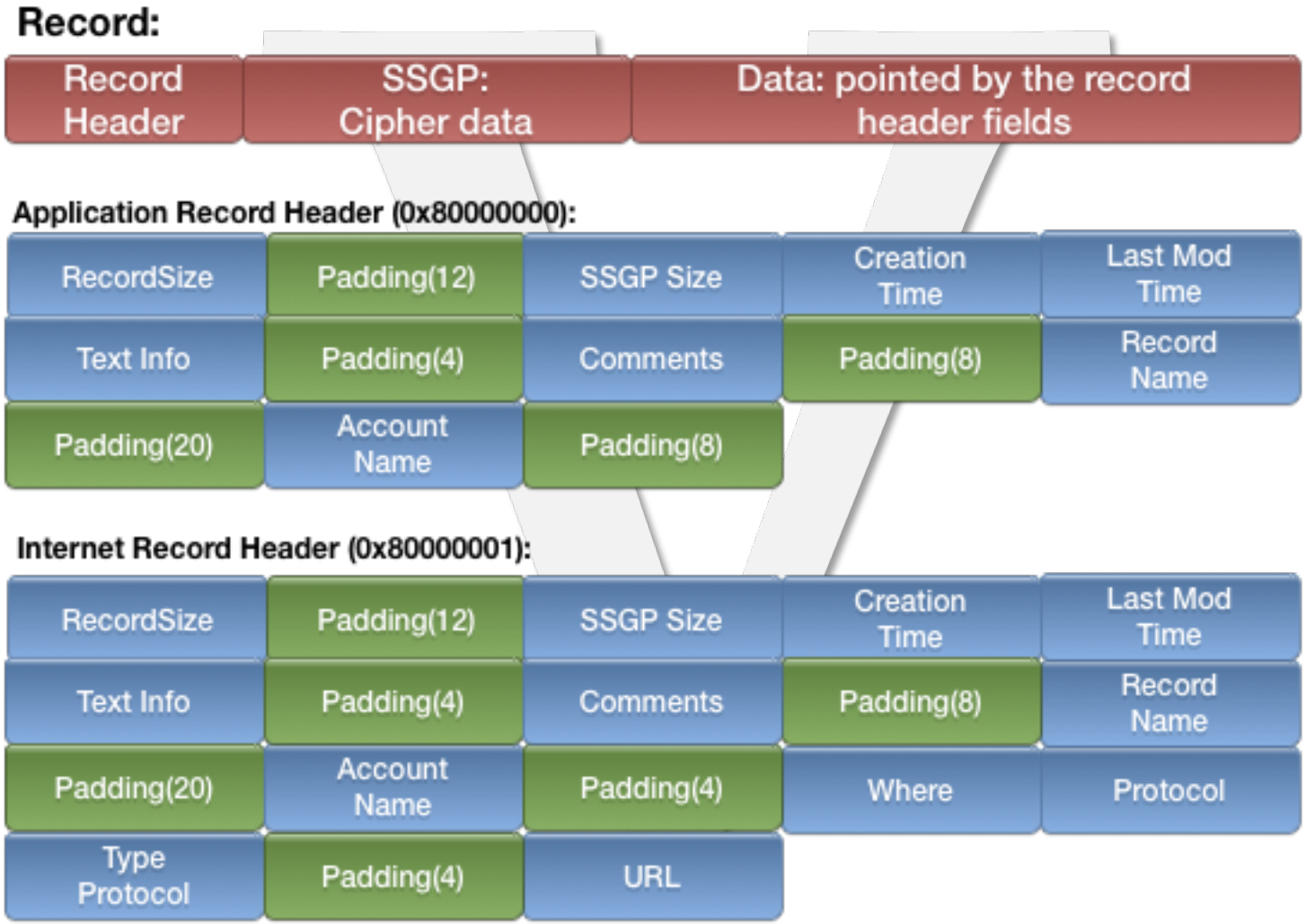


Keychain

- Timestamp en texto y estructura en big endian.
- Sistema: /Library/Keychains/System.keychain
- Usuario: $home/Library/Keychains/login.keychain
- Grupo de tablas donde cada tabla almacena un tipo de dato o record:
  - Application record: aplicaciones, notas y wireless.
  - Internet record: navegadores, correos y otros.
- Plaso: mac_keychain.py


Keychain Database:
DB HEADER
DB SCHEMA
TABLE 1
TABLE N
DB Password
DB Header:
Magic
Version
Header Size
Schema Offset
Padding
DB Schema:
Schema Size
Number Tables
Table1 Offset
Table2 Offset
TableN Offset
Table 1:
Table Size
Record Type
Number Records
First Record
Index Offset
Padding
Records:
Record 1
Record 2
Record 3
Record N
Index
Table N:
Table Size
Record Type
Number Records
First Record
Index Offset
Padding
Records:
Record 1
Record 2
Record 3
Record N
Index

Tablas Application y Internet:



Document Versions

- En 10.7 nueva funcionalidad para disponer de versiones anteriores de un fichero.
- Cuando se guarda un fichero Versions hace una copia de esa versión.
- Guarda: /.DocumentRevisions-V100/PerUID/UserUID/
- Gestiona: /.DocumentRevisions-V100/db-V1/db.sqlite
- Propio de cada unidad (partición).
- Plaso: mac_document_versions.py

Document Versions

- Fecha de cuando se guardó la copia.
- Path real del fichero.
- Path a la copia del fichero.
- UID del usuario que guardo el documento.

```
SELECT f.file_name AS original_name, f.file_path AS original_path,
f.file_last_seen AS last_time_checked, g.generation_path AS version_path,
g.generation_add_time AS version_time
FROM files f, generations g
WHERE f.file_storage_id = g.generation_storage_id;
```

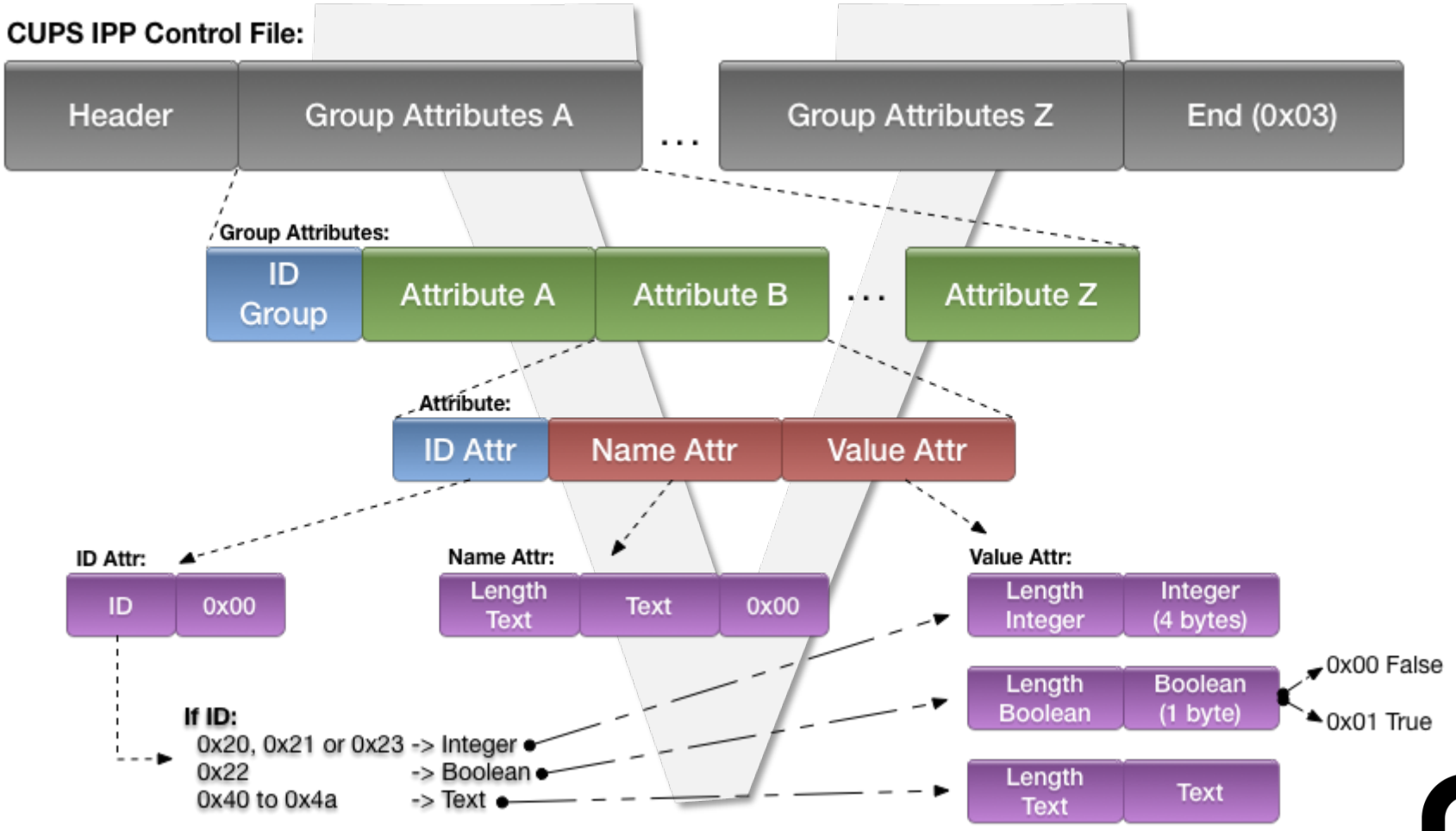
Impresoras remotas: CUPS IPP

- Log de los trabajos impresos remotamente.
- Big endian y timestamp en Epoch.
- Mac emplea IPP 2.0 (RFC2910 es 1.1).
- Timestamp no son tipo date, sino enteros.
- Donde: /private/var/spool/cups/
- Un fichero cXXXXX por trabajo impreso.
- Plaso: cups_ipp.py

CUPS IPP: ¿qué aporta?

- El nombre del trabajo y impresora.
- Programa que lo imprimió y tipo fichero.
- Nombre completo y usuario que lo imprimió.
- Cuando fue impreso, cuando fue procesado y cuando termino de imprimirse.
- **NO** sabemos el nombre del documento, pero sí se imprimió usando el driver PostScript se crea una copia del documento original en el mismo directorio:
  - Ejemplo: IPP c00026, Copia d00026

CUPS IPP


CUPS IPP Control File:
Header
Group Attributes A
...
Group Attributes Z
End (0x03)
Group Attributes:
ID Group
Attribute A
Attribute B
...
Attribute Z
Attribute:
ID Attr
Name Attr
Value Attr
ID Attr:
ID
0x00
Name Attr:
Length Text
Text
0x00
Value Attr:
Length Integer
Integer (4 bytes)
Length Boolean
Boolean (1 byte)
0x00 False
0x01 True
Length Text
Text
If ID:
0x20, 0x21 or 0x23 -> Integer
0x22 -> Boolean
0x40 to 0x4a -> Text

CUPS IPP: Campos importantes

Name Attribute	Value
time-at-creation	When the job was created
time-at-processing	When the job starts to be processed
time-at-completed	When the job was finished
printer-uri	The printer URL
copies	The number of document copies
job-originating-user-name	System user name
job-name	Name of the job
document-format	Type of document
job-originating-host-name	Computer name
com.apple.print.JobInfo.PMApplicationName	Application used to print
com.apple.print.JobInfo.PMJobOwner	Completed name of the user

Lista de propiedades (Plist)

- Gran número de ficheros repartidos por el sistema.
- Información relevante, configuraciones, timestamp (Cocoa), passwords, etc.
- Información guardada en formato XML(plistlib) o binario (binplist).
- Son estructuras que almacenan datos básicos.

Demasiado bonito

- Las estructuras Plist pueden contener como atributos otras estructuras Plist.
- Algunos tipos son estructuras binarias empleadas por los frameworks cuyo código fuente o documentación no está disponible.
- Librerías y XCode no tiene en cuenta lo anterior.
- Veremos los Plist más relevantes.

Plist: Usuarios del sistema

- /private/var/db/dslocal/nodes/Default/users/name
- El Plist contiene el nombre del usuario, UID, GUID, home, nombre completo, shell, etc. Y dos importantes atributos:
- Passwordpolicyoptions (Plist XML)
  - passwordLastSetTime
  - lastLoginTimestamp
  - failedLoginTimestamp y failedLoginCount
- ShadowHashData (Plist binaria, ¡Pentesters!)

Plist: ShadowHashData

- Algoritmo PBKDF2-SHA-512 (Mac 10.8 y 10.9)
- La estructura contiene 3 atributos:
  - Iterations: número de iteraciones.
  - Salt: sal de la contraseña.
  - Entropy: hash del password (256 hex char)
- Recuperación de password lento (15 a 35/s en i7):
  - dave -p fichero.plist [opciones]
  - haschat –m7100 hash.txt [opciones] (+rápido)
    - mliteracionessalentropia(solo los primeros 128)
    - Bug con x86, reportado y solucionado en próxima versión.

Plist: siempre quedará el autologin

- /Library/Preferences/com.apple.loginwindow.plist
- Si atributo *autoLoginUse*r existe, el nombre que contiene es el nombre del usuario que tiene activado el autologin.
- Password almacenado en */etc/kcpassword* haciendo Xor a la magic key *0x7d895223d2bcddeaa3b91f*.

```
# xxd /etc/kcpassword
0000000: 1ceb 3147 d217 2f11 40ff 63bf        ..1G../.@.c.
# python kcpass.py 1ceb3147d2172f1140ff63bf

        Kcpasswd: 0x1ceb3147d2172f1140ff63bf.
        Magic Xor: 0x7d895223d2bcddeaa3b91f.
        The password is: "abcd".

#
```

Plist: Timemachine y Bluetooth

- /Library/Preferences/com.apple.TimeMachine.plist
  - Disco ID donde se realiza la copia de seguridad.
  - Timestamp de todas las copias.
  - BackUpAlias: byte 11 tamaño texto seguido del texto.

- /Library/Preferences/com.apple.Bluetooth.plist
  - Información de los dispositivos Bluetooth.
  - Si se asociaron con el sistema y cuando.

Plist: actualizaciones, wireless y FW

- /Library/Preferences/com.apple.SoftwareUpdate.plist
  - Versión instalada y fecha última actualización.
  - Actualizaciones y versiones pendientes.
- /Library/Preferences/SystemConfiguration/com.apple.airport.preferences.plist
  - Configuración de las WiFi.
  - Última vez que nos conectamos a esa WiFi.
- /Library/Preferences/com.apple.alf.plist
  - Configuración del firewall (no timestamps)

Plist: spotlight, histórico y Apple

- /Users/user/Library/Preferences/com.apple.spotlight.plist
  - Término buscado, elemento asociado y timestamp.
- /Library/Receipts/InstallHistory.plist
  - Histórico de instalaciones de programas.
  - Nombre, versión, como fue instalado y cuando.
- /Users/user/Library/Preferences/ByHost/com.apple.coreservices.appleidauthenticationinfo.APPLE-UUID.plist
  - Información de la cuenta Apple del usuario.
  - Última vez que la cuenta fue usada.

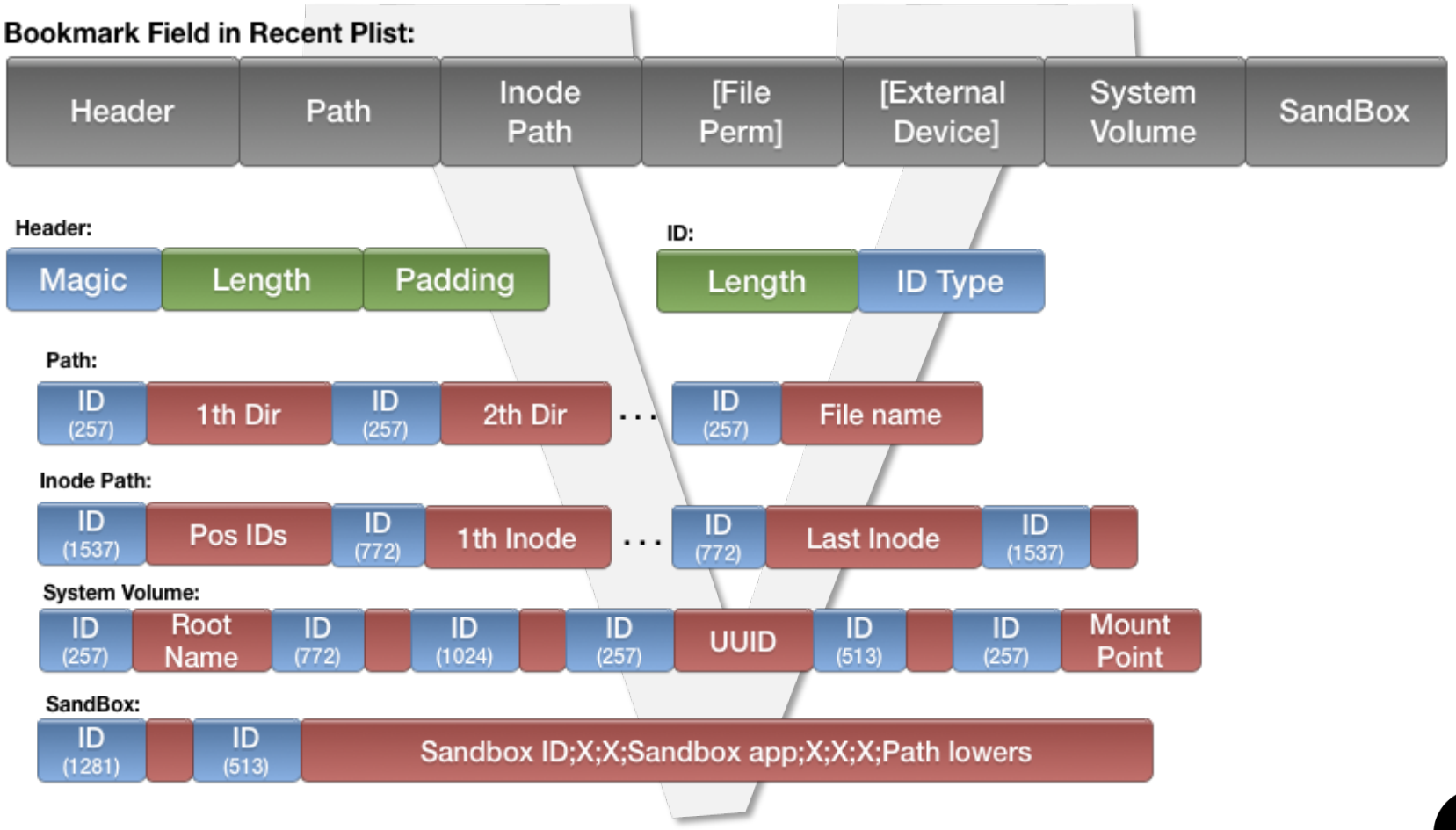
Plist: todo soportado en Plaso

- airport.py
- appleaccount.py
- bluetooth.py
- install_history.py
- macuser.py
- softwareupdate.py
- spotlight.py
- spotlight_volume.py
- timemachine.py

Más Plist! Recent files...

- /Users/user/Library/Preferences/
  - com.apple.recentitems.plist
  - *.LSSharedFileList.plist
- Lista de elementos donde cada elemento tiene:
  - Name: nombre del fichero.
  - Bookmark: estructura binaria en little endian del framework LSS.
    - Path del fichero (primera vez)
    - Path de inodos (actualizado)
    - Partición desde donde se leyó el fichero

Plist: recent files



Plist: recent files, ejemplo...

- POC: mac_recent.py

```
DarkTemplar:recent moxilo$ python mac_recent.py ejemplos/org.niltsh.MPlayerX.LSShar
edFileList.plist
File: ejemplos/org.niltsh.MPlayerX.LSSharedFileList.plist

        Recent document open by MPlayerX(niltsh): Resident Evil 2002.mkv
        Path: /Volumes/Peliculas/Peliculas HD/ResidentEvil/Resident Evil 2002.mkv
        Inode Path: /12356/2/19/28/37
        External device: /Volumes/Peliculas
        HD Partition Root Name: Macintosh HD
        HD Root UUID: 43B7DEF7-8F02-3A55-820A-2F4DE404F33A
        HD Root mount in: /
```

- PD: no es lo que parece, son las vacas de SPaNKeR ...

Último Plist! Sidebar

- /Users/usr/Library/Preferences/com.apple.sidebarlists.plist
- Lista de elementos donde en cada lista hay dos atributos:
  - Name: nombre a visualizas en Finder.
  - Alias: estructura binaria en big endian y timestamp HFS si partición es HFS/DMG o "0" de lo contrario.
- Systemitems: histórico de dispositivos montados.
  - Mac OS 10.9 ficheros DMG no se tienen en cuenta.
  - Los timestamp son los correspondientes a la imagen.

Plist: sidebard, ejemplo

- POC: alias.py

```
Name: BackUp2
Time: 2013-09-14 21:40:56
Volume name: BackUp2
Volume time: 2013-09-14 21:40:56
Mount point: /Volumes/BackUp2

Name: Peliculas
Time: 1904-01-01 00:00:00
Volume name: Peliculas
Volume time: 1904-01-01 00:00:00
Mount point: /Volumes/Peliculas

Name: ZERATUL
Time: 1904-01-01 00:00:00
Volume name: ZERATUL
Volume time: 1904-01-01 00:00:00
Mount point: /Volumes/ZERATUL

Name: Wireshark
Time: 2013-09-10 18:01:29
Volume name: Wireshark
Volume time: 2013-09-10 18:01:29
Mount point: /Volumes/Wireshark
```

¿Se acabó? Hay mucho más...

- Ficheros .DS_Store de Finder tienen diversas funciones como recuperar ficheros de .Trash.
- Swap, Hybernation File, extensiónes del kernel.
- Caches: estructuras binarias únicas.
- Ficheros out (/private/var/log/*.out)
- Y por supuesto, ¡más Plists!
- Todo estará en Plaso para antes de verano ☺

¿Preguntas?


AN UPDATE IS AVAILABLE FOR YOUR COMPUTER
stickycomics.com
COOL, MORE FREE STUFF!
NOT AGAIN!
OOH, ONLY $99!
linux
windows
mac

¡Gracias!