Forensic Analysis of the Windows Registry in Memory

By

Brendan Dolan-Gavitt

Presented At

The Digital Forensic Research Conference

DFRWS 2008 USA Baltimore, MD (Aug 11th - 13th)

Forensic Analysis of the Windows Registry in Memory

Brendan Dolan-Gavitt

What’s the registry?

- Centralized database that stores configuration information for Windows systems.
- Appears as a single hierarchy to the OS, but is actually made up of separate *hive* files unified into a single namespace.

What’s the registry? (cont.)

- Organized into *keys* and *values*. Keys are somewhat like directories, and can contain subkeys or values.
- Values are strongly typed: `REG_SZ` (string), `REG_DWORD` (integer), etc.

Why the registry?

- Lots of forensically useful information!
- Recently run programs, recent wireless networks, devices recently attached to the system (eg, USB keys)
- All keys are timestamped when written
- Harlan Carvey has done a lot of work in this area. (RegRipper)

The registry in memory

- Subsystem called the *Configuration Manager* loads hives into memory, places them into unified namespace.
- Keys and values link to one another using *cell indexes*, which are essentially pointers to other locations in the hive.
- Key and value data can be *stable* (flushed out to the on-disk hive), or *volatile* (dynamically generated, only in memory).

Cell indexes

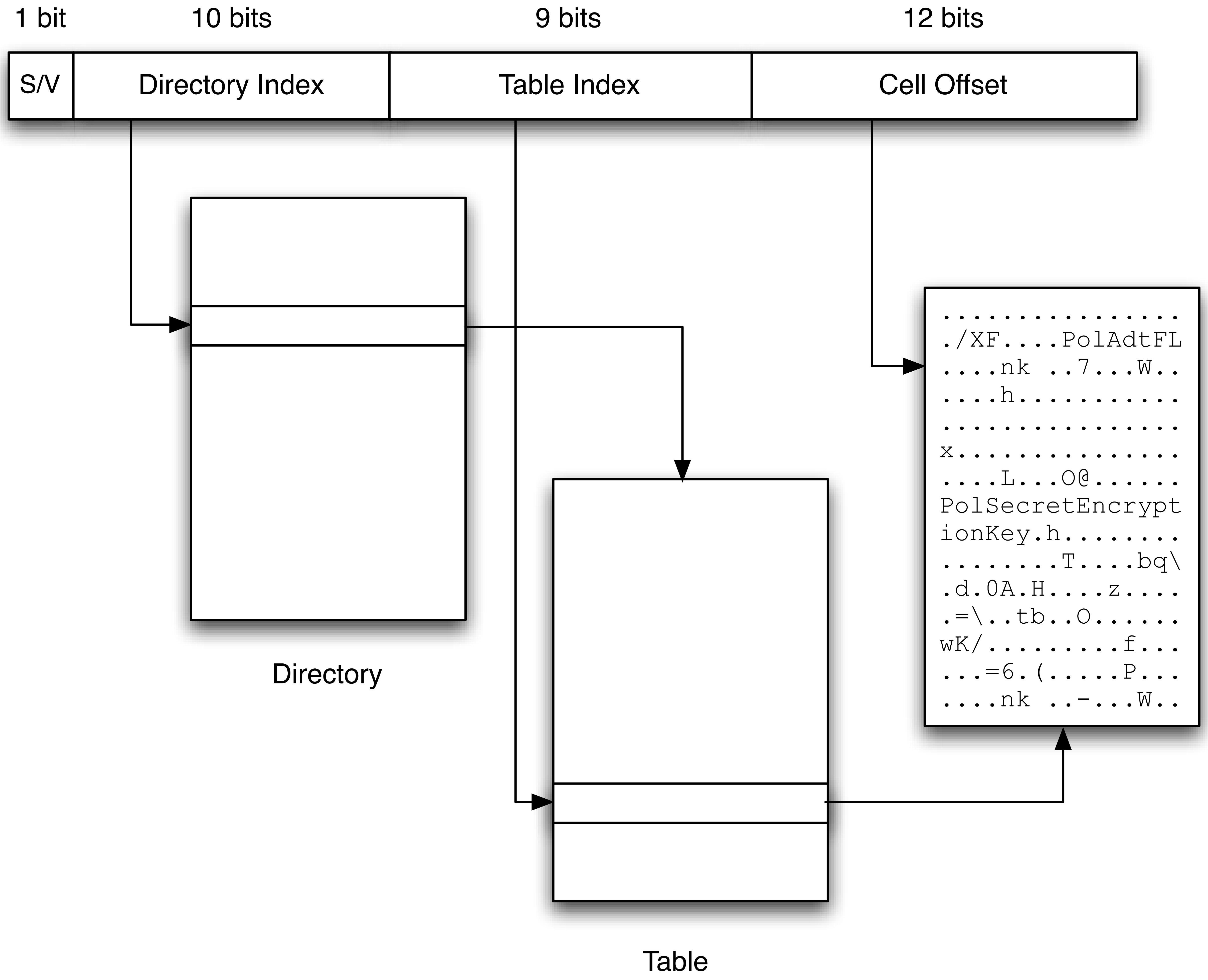
- On disk, cell indexes are mapped to file offsets using the formula `offset = ci + 0x1000`
- In memory, more complicated: each index must be translated into a virtual address.
- To do this, we use a mapping table stored in the data structure representing a hive.

Finding the hives

- Data structure that represents a registry hive in memory: `_CMHIVE`
- Handy signature: `0xbee0bee0`
- Pool tag: `CM10`
- Once one is found, we can use the kernel address space and list-walk to find the others!

Cell index translation

- Very similar to x86 non-PAE address translation!
- Cell index is divided into pieces, which give offsets into the mapping tables.
- Array of 2 `_HMAP_DIRECTORY` (found in `_CMHIVE.Hive.Storage`)
  - Array of `_HMAP_TABLE` structures (1024)
    - Array of `_HMAP_ENTRY` structures (512)
- Entries can be 0, meaning that the data has not been brought in from disk.



```
................
./XF....PolAdtFL
....nk ..7...W..
....h...........
................
x...............
....L...O@......
PolSecretEncrypt
ionKey.h........
........T....bq\
.d.0A.H....z....
.=\..tb..O......
wK/.........f...
...=6.(.....P...
....nk ..-...W..
```

Accessing keys and values

- Once cell index differences are accounted for, works just like on disk
- Exception: each key can have both stable and volatile subkeys.
- Within key structure, member `SubKeyLists` is actually an array of length 2
- Most non-MS registry parsers treat the second entry as “unknown”

How much data is in memory?

- Depends on level of system activity
- For lightly loaded systems (test VM, NIST images) over 98% of hive data was recoverable.
- For more heavily loaded systems, much less (around 50%)
- Depends whether the data you want has been used recently.

Volatile Data Examples

- Hardware description
- Mounted volumes
- Computer name
- User environment

Attacking cached data

- In the registry, data is flushed from memory back to disk every 5 seconds (Russinovich, 2004)
- However, if attacker bypasses normal update mechanisms and writes to memory directly, data may not get flushed!
- An attacker can use this to alter the runtime configuration of the machine without leaving traces on disk

Example: Changing the admin password

- Password hashes are stored in registry.
- Find the location in memory corresponding to the hash in the registry
- Change it to a precomputed value like HASH(“foobar”) (harder than it sounds)

Example: Changing the admin password (cont.)

- Log out so that the LSA subsystem will pick up the change
- Log in with your new password!
- Upon reboot, everything is back to normal: old password works, no trace on disk.

Detecting the attack

- Since we can read the registry directly from memory, no problem
- Extract registry from memory, and compare to disk.
- If things don't match, could indicate data altered only in memory.

Implementation

- Implemented using Volatility Framework (new HiveAddressSpace handles cell index translation)
- Code currently only works with Volatility 1.1.1 + heavy local modifications
- Work underway to port to Volatility 1.3, release as open source

Finding Hives

```
$ ftimes --diglean cmhive.ft xp-laptop-2005-07-04-1430.img
name|type|tag|offset|string
xp-laptop-2005-07-04-1430.img|normal||42168322|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||42195802|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||47598386|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||155764586|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||155973602|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||208587610|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||208964442|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||234838874|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||243852930|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||251418754|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||252887042|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||256039730|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||269699930|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||339523202|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||346659674|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||377572186|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||387192178|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||509150850|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||521194330|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||523667586|%95%0cCM10%e0%be%e0%be
xp-laptop-2005-07-04-1430.img|normal||527756082|%95%0cCM10%e0%be%e0%be
```

Listing registry hives

```
$ ./volatility hivelist -o 42168328 -f xp-laptop-2005-07-04-1430.img
0xe2610b60L \Documents and Settings\[...]\UsrClass.dat
0xe25f0578L \Documents and Settings\[...]\NTUSER.DAT
0xe1d33008L \Documents and Settings\LocalService\[...]\UsrClass.dat
0xe1c73888L \Documents and Settings\LocalService\NTUSER.DAT
0xe1c04688L \Documents and Settings\NetworkService\[...]\UsrClass.dat
0xe1b70b60L \Documents and Settings\NetworkService\NTUSER.DAT
0xe1658b60L \WINDOWS\system32\config\software
0xe1a5a7e8L \WINDOWS\system32\config\default
0xe165cb60L \WINDOWS\system32\config\SAM
0xe1a4f770L \WINDOWS\system32\config\SECURITY
0xe1559b38L
0xe1035b60L \WINDOWS\system32\config\system
0xe102e008L
```

Showing arbitrary keys and values

```
$ ./volatility printkey -f xp-laptop-2005-07-04-1430.img -o 0xe25f0578 \
   'Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2'

Key name: MountPoints2 (Stable)
Last updated: Mon Jul  4 14:18:04 2005

Subkeys:
   C (Stable)
   D (Stable)
   {47c255f0-e599-11d9-b395-000625abeee3} (Stable)
   {6a2b71c4-9e1a-11d8-b4c2-806d6172696f} (Stable)
   {d95794c1-9e1f-11d8-b2ac-806d6172696f} (Stable)
   CPC (Volatile)

Values:

$ ./volatility printkey -f xp-laptop-2005-07-04-1430.img -o 0xe25f0578 \
   'Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2\CPC\Volume'

Key name: Volume (Volatile)
Last updated: Mon Jul  4 14:18:04 2005

Subkeys:
   {47c255f0-e599-11d9-b395-000625abeee3} (Volatile)
   {6a2b71c4-9e1a-11d8-b4c2-806d6172696f} (Volatile)

Values:
```

Extracting Password Hashes

```
$ ./volatility hashdump -f xp-laptop-2005-07-04-1430.img \
   --sys-offset 0xe1035b60 --sam-offset 0xe165cb60

Administrator:500:08f3a52bdd35f179c81667e9d738c5d9:ed88cccbc08d1c18bcded317112555f4:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:ddd4c9c883a8ecb2078f88d729ba2e67:e78d693bc40f92a534197dc1d3a6d34f:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:8bfd47482583168a0ae5ab020e1186a9:::
*******:1003:07b8418e83fad948aad3b435b51404ee:53905140b80b6d8cbe1ab5953f7c1c51:::
ASPNET:1004:2b5f618079400df84f9346ce3e830467:aef73a8bb65a0f01d9470fadc55a411c:::
*****:1006:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

Future work

- Support for Windows 2000 and Vista
- Try to reconstruct tree even if some links are missing.
- Automate on-disk vs. memory comparisons

Thanks for listening!

- You can find me at:
  - brendandg@gatech.edu
  - http://moyix.blogspot.com/
  - irc.freenode.com #volatility as moyix
- Questions?