Beyond files forensic
OWADE cloud based forensic

Elie Bursztein  Stanford University
Ivan Fontarensky  Cassidian
Matthieu Martin  Stanford University
Jean Michel Picod  Cassidian
The world is moving to the cloud
2.7 millions photos are uploaded to Facebook every 20 minutes
100 millions new files are saved on Dropbox every day
Data are moving to multiple services

Hard drive
Data are moving to multiple services

Hard drive → emails
Data are moving to multiple services

Hard drive → emails → Cloud
Data are moving to multiple services

- Hard drive
- Emails
- Cloud
- Webmail
Data are moving to multiple services

- Hard drive
- Emails
- Contacts
- Cloud
- Webmail

Beyond files recovery: OWADE cloud based forensic

http://owade.org
Data are moving to multiple services

Hard drive → emails → contacts

Cloud → Webmail → Social sites

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin
Beyond files recovery: OWADE cloud based forensic
http://owade.org
Data are moving to multiple services

Hard drive → emails → contacts → photos

Cloud → Webmail → Social sites
Data are moving to multiple services

Hard drive →
- emails
- contacts
- photos

Cloud →
- Webmail
- Social sites
- Photo sites
There are more data which are harder to reach.

Dealing with cloud data force us to reinvent forensic.
Let’s do cloud forensics
What is **cloud forensics**?
Facebook credentials as a use case
Facebook credentials as a use case

IE
DPAPI
Blob

credentials

Facebook

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin
Beyond files recovery: OWADE cloud based forensic

http://owade.org
Facebook credentials as a use case
Facebook credentials as a use case
Facebook credentials as a use case

Syskey → Registry

Windows User Password → SAM (hash)

DPAPI master-key → DPAPI blob-key

IE DPAPI Blob → credentials

Facebook

Registification credentials via:
- SAM (hash)
- DPAPI master-key
- IE DPAPI Blob
Facebook credentials as a use case

Getting Facebook credentials require to **bypass 4 layers of encryption**
Show you how to **bypass the encryption layers** and get the data you want
Introducing OWADE

• Dedicated to cloud forensics
• Decrypt / recovers
  • DPAPI secrets
  • Browsers history and websites credentials
  • Instant messaging creds
• Wifi data
• Free and open-source

http://owade.org
Beyond files recovery: OWADE cloud based forensic

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin

http://owade.org
OWADE overview
disk
OWADE overview

disk

Registry

disk image

http://owade.org
Beyond files recovery: OWADE cloud based forensic

- Disk
- Disk image
- Registry
- Files

OWADE overview

http://owade.org
OWADE overview

- Disk
- Disk image
- Registry
- Files
- Windows credentials
Beyond files recovery: OWADE cloud based forensic

- Disk
- Disk image
- Registry
- Windows credentials
- Files
- WiFi info

http://owade.org
OWADE overview

- Disk
- Disk image
- Registry
- Windows credentials
- WiFi info
- Files
- Hardware info
Beyond files recovery: OWADE cloud based forensic

- Disk
- Disk image
- Registry
- Windows credentials
- WiFi info
- Files
- Hardware info
- Credentials and data

http://owade.org
Beyond files recovery: OWADE cloud based forensic

- Disk
- Disk image
- Registry
- WiFi info
- Files
- Hardware info
- Windows credentials
- Cloud data
- Credentials and data

http://owade.org
• File base forensics refresher
Outline

• File base forensics refresher
• The Windows crypto eco-system
Outline

• File base forensics refresher
• The Windows crypto eco-system
• Wifi data and Geo-location
Outline

• File base forensics refresher
• The Windows crypto eco-system
• Wifi data and Geo-location
• Recovering browser data
Outline

- File base forensics refresher
- The Windows crypto eco-system
- Wifi data and Geo-location
- Recovering browser data
- Recovering instant messaging data
Outline

• File base forensics refresher
• The Windows crypto eco-system
• Wifi data and Geo-location
• Recovering browser data
• Recovering instant messaging data
• Acquiring cloud data
Outline

• File base forensics refresher
• The Windows crypto eco-system
• Wifi data and Geo-location
• Recovering browser data
• Recovering instant messaging data
• Acquiring cloud data
• Demo
File based forensic refresher
### Not all files are born equal

<table>
<thead>
<tr>
<th>Type of file</th>
<th>how to recover it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>copy</td>
</tr>
<tr>
<td>In the trash</td>
<td>undelete utility</td>
</tr>
<tr>
<td>Deleted</td>
<td>file carving</td>
</tr>
<tr>
<td>Wiped</td>
<td>call the NSA :)</td>
</tr>
</tbody>
</table>

[http://owade.org](http://owade.org)
Windows registry

- .dat files
- Hardware information
- Softwares installed with their versions and serials
- Windows credentials (encrypted)
Some Registry Information Extracted
Windows crypto
Why do we care about Windows crypto?
The Windows crypto eco-system

Crypto API
The Windows crypto eco-system

Crypto API

SAM

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin

Beyond files recovery: OWADE cloud based forensic

http://owade.org
Beyond files recovery: OWADE cloud based forensic

http://owade.org
The Windows crypto eco-system

Crypto API

SAM

DPAPI

Credential Manager

http://owade.org
Windows Crypto API

- Basic cryptographic blocks
  - Cipher: 3DES, AES
  - Hash functions: SHA-1, SHA256, HMAC
  - PKI: public keys and certificates (X.509)
The Security Account Manager (SAM)

- Store Windows user credentials
- Located in the registry
- Encrypted with the SYSKEY
- Passwords are hashed
Windows Password Hashing functions

- Two hash functions used
  - LM hash function (NT, 2K, XP, VISTA) weak
  - NTLM (XP, Vista, 7)
- Passwords are not salted
LM hash weakness

- Use only upper-case
- Hash password in chunk of 7 characters

mypassword $\rightarrow$ LMHash(MYPASSW) + LMHash(ORD)

Password key-space: $69^7$ (at most)
Rainbow Tables

• Pre-compute all the possible passwords
• Time-Memory trade-off
• Rainbow tables of all the LM hash are available
How OWADE Works

• Extract Usernames and password hashes
• LM hashes available?
  • use John/Rainbow tables to get the pass in uppercase
  • use NTLM hashes to find the password cases
• Try to crack the NTLM using John/Rainbow table
Windows Password recovered
If the password is too strong we *can't recover it*
but we can **still decrypt** DPAPI secret (sometime)
The Data Protection API

- Ensure that encrypted data can’t be decrypted without knowing the user Windows password

- Blackbox crypto API for developers:
  - Encrypt data → DPAPI blob
  - Decrypt DPAPI blob → data

- Main point: tie the encryption to the user password
DPAPI derivation scheme

User → SHA1(password) → pre-key
DPAPI derivation scheme

User

\[\text{SHA1(password)}\]

pre-key

master-key
DPAPI derivation scheme

User

\[ \text{SHA1(\text{password})} \rightarrow \text{pre-key} \]

\[ \text{master-key} \]

\[ \text{blob key} \]
DPAPI derivation scheme

User

\[\text{SHA1(password)}\]

pre-key

master-key

blob key

DPAPI blob
DPAPI derivation scheme

User → SHA1(password) → pre-key → master-key → blob key → DPAPI blob

SHA1(password) is derived from the password using SHA1 hash function.
Pre-key is used to generate master-key.
Master-key is used to generate three different blob keys, each encrypted with a DPAPI blob.
DPAPI Blob structure

```c
struct wincrypt_datablob {  
    DWORD cbProviders,
    GUID pbProviders[cbProviders],
    DWORD cbMasterkeys,
    GUID pbMasterkeys[cbMasterkeys],
    DWORD dwFlags,
    DWORD cbDescription,
    BYTE pbDescription[cbDescription],
    ALG_ID algCipher,
    DWORD cbKey,
    DWORD cbData,
    BYTE pbData[cbData],
    DWORD dwUnknown,
    ALG_ID algHash,
    DWORD dwHashSize,
    DWORD cbSalt,
    BYTE pbSalt[cbSalt],
    DWORD cbCipher,
    BYTE pbCipher[cbCipher],
    DWORD cbCrc,
    BYTE pbCrc[cbCrc]
} ;
```
struct wincrypt_masterkey_masterkeybloc
{
    DWORD    dwRevision,
    BYTE     pbSalt[16],
    DWORD    dwRounds,
    ALG_ID   algMAC,
    ALG_ID   algCipher,
    BYTE     pbEncrypted[
};
DPAPI blob
DPAPI blob \rightarrow \text{Master-key GUID}
DPAPI blob → Master-key GUID → Master key → pre-key
DPAPI blob → Master-key GUID

Master key → pre-key

User

SHA1(password) → pre-key
Beyond files recovery: OWADE cloud based forensic

- DPAPI blob
  - Master-key GUID
  - Master key
    - pre-key
      - SHA1(password)
        - User

E. Bursztein, I. Fontaresky, J.M. Picod, M. Martin
DPAPI blob

Master-key GUID

Master key

Cipher + key

pre-key

Master key

SHA1(password)

User
DPAPI blob → Master-key GUID

Master-key GUID → Master key

Master key → pre-key

pre-key → cipher + key

Cipher + key → SHA1(password)

User

SHA1(password) → Master key

Master key → blob key

blob key
DPAPI additional entropy

- Software can supply an additional entropy
  - Act as a “key” (needed for decryption)
  - Force us to understand how it is generated for each software
  - Can be used to tie data to a specific machine (i.e. Netbios name)
• If we can’t crack the password we need its SHA1
• This SHA1 is stored in the hibernate file
• OWADE uses Moonsols to recover it
Credential Manager

- Built on top of DPAPI
- Handle transparently the encryption and storage of sensitive data
- Used by Windows, Live Messenger, Remote desktop...
## Credstore type of credentials

<table>
<thead>
<tr>
<th>Type of credential</th>
<th>Encryption</th>
<th>Example of application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic password</td>
<td>DPAPI + fixed string</td>
<td>Live messenger HTTP auth (IE)</td>
</tr>
<tr>
<td>Domain password</td>
<td>In clear</td>
<td>Netbios</td>
</tr>
<tr>
<td>Domain certificate</td>
<td>Hash of certificate</td>
<td>Certificate</td>
</tr>
<tr>
<td>Domain visible password</td>
<td>DPAPI + fixed string</td>
<td>Remote access .NET passport</td>
</tr>
</tbody>
</table>
WiFi data
Wifi data

- Info stored for each access point
  - Mac address (BSSID)
  - Key (encrypted)
  - Last time of access
- Wifi data are stored in
  - Registry (XP)
  - XML file and Registry (Vista/7)
Decoding WiFi password

- Encrypted with DPAPI
- Access point shared among users
  - Encrypted with the System account
  - But the system account has no password...

What is my DPAPI key??
Decrypting WiFi password

- Use a LSASecret as DPAPI key
- Array of credentials
  - HelpAssistant password in clear
  - DPAPI_SYSTEM
  - “Encrypted”
Where are you?

- We’ve recovered access point keys but where are they?
• We’ve recovered access point keys but where are they?

There is an app for that!
Location-Aware Browsing

Firefox can tell websites where you’re located so you can find info that’s more relevant and more useful. It’s about making the Web smarter – and is done in a way that totally respects your privacy. Give it a try!

Frequently Asked Questions

- What is Location-Aware Browsing?
- How does it work?
- How accurate are the locations?
- What information is being sent, and to whom? How is my privacy protected?
- Am I being tracked as I browse the web?
HTML5 Geo-location protocol

Location-Aware Browsing

Firefox can tell where you are, and you can find information that is done in a way that respects privacy. Give it a try.

Frequently Asked Questions

- What is Location-Aware Browsing?
- How does it work?
- How accurate are the locations?
HTML5 Geo-location protocol

Location-Aware Browsing

Firefox can tell you where you are. You can find information that's useful. It's about sharing your location in a way that preserves your privacy. Give it a try.

Frequently Asked Questions

+ What is Location-Aware Browsing?
+ How does it work?
+ How accurate are the locations?
Google Wi-Fi Data Collection Angers European Officials

Brennon Slattery, PC World  May 17, 2010 7:08 am

European officials are still miffed over Google's "accidental" Wi-Fi data collection and seek an in-depth investigation that may lead to harsh penalties for the search engine giant.

It was revealed that Google's Street View cars were collecting more than images and coordinates for its sophisticated GPS site. As much as 600GB of data from Wi-Fi networks -- in more than 30 countries -- has been snagged in Google's fishnet.
Nothing is ever easy

• Google started to restrict queries in June
• So we started to look for other API
Entering Microsoft

• Live service
• “Documented” in the Windows mobile MSDN
• After sniffing the traffic:
  • Use a big SOAP request
  • Does not check any ID fields
  • Allows to supply one MAC

<GetLocationUsingFingerprint xmlns="http://inference.location.live.com">
  <RequestHeader>
    <Timestamp>2011-02-15T16:22:47.0000968-05:00</Timestamp>
    <ApplicationId>e1e71f6b-2149-45f3-b298-a20xxxxxxxx5017</ApplicationId>
    <TrackingId>21BF9AD6-CFD3-46B2-B042-EE90xxxxxxxx</TrackingId>
    <DeviceProfile ClientGuid="0fc571be-4622-4ce0-b04e-xxxxxxxeb1a222" Platform="Windows7" DeviceType="PC"
OSVersion="7600.16695.amd64fre.win7_gdr.101026-1503"
LFVersion="9.0.8080.16413" ExtendedDeviceInfo="" />
    <Authorization />
  </RequestHeader>
  <BeaconFingerprint>
    <Detections>
      <Wifi7 BssId="00:BA:DC:0F:FE:00" rssi="-25" />
    </Detections>
  </BeaconFingerprint>
</GetLocationUsingFingerprint>
Using the Microsoft Geolocalization API to retrace where a Windows laptop has been

July 29, 2011 | Privacy

EDIT (Tuesday 2nd August) Microsoft Statement is available from here

EDIT (Sunday 31st July) The flaw is fixed! I had a phone call with some people from Microsoft yesterday (yes on a Saturday) and they told me they fixed the problem. I will update this post with their response as soon as it is out. The demo code does not work anymore.

In our upcoming BlackHat talk, we will show you how the WiFi data stored by Windows can be used to geolocate where your computer has been. While the ability to retrace where a computer has been (and when) certainly carries privacy implications, in this post I want to focus on how we uncovered this data, and the unexpected difficulties we encountered while developing this technique.
• Fixed last weekend
• No longer return location for a single address
Geo-location API restrictions

Requires 2 MAC close from each other

The MAC and IP location need to be “close”

Requires multiples MAC addresses

see [http://elie.im/blog/](http://elie.im/blog/) for more information
WiFi Information Extracted By OWDE
Browsers
Firefox > 3.4

- Passwords
  - Location: `signons.sqlite`
  - Encryption: 3DES + Master password

- History
  - URLs: `places.sqlite`
  - Forms fields: `formhistory.sqlite`
Decrypting Firefox password
Decrypting Firefox password
Decrypting Firefox password

User → pass → Global salt

key3.db
Decrypting Firefox password

User → pass → key3.db

User key: HMAC-SHA1(salt, pass)

Global salt
Decrypting Firefox password

User → pass → user key: HMAC-SHA1 (salt, pass) → Global salt → encrypted key + key salt → key3.db

http://owade.org
Decrypting Firefox password

User → pass → user key: HMAC-SHA1 (salt, pass) → Global salt → encrypted key + key salt → master key: 3DES(userkey, enckey) → key3.db
Decryption of Firefox password:

1. User enters password (pass).
2. pass is hashed to create a user key: $\text{HMAC-SHA1}(\text{salt}, \text{pass})$.
3. The user key is encrypted using 3DES: $\text{3DES}($userkey, enckey$)$.
4. The encrypted key and key salt are saved in key3.db.
5. The master key is created: $\text{3DES}($userkey, enckey$)$.
6. The encrypted password is stored in signon.sqlite.

Key3.db contains the encrypted key and key salt, allowing the master key to be reconstructed.
Decrypting Firefox password

User → pass → Global salt

user key: HMAC-SHA1(salt, pass)

down → encrypted key + key salt

master key: 3DES(userkey, enckey)

down → encrypted pass

Site password: 3DES(master key, enc pass)

key3.db

signon.sqlite
## Every form field is recorded

<table>
<thead>
<tr>
<th>id</th>
<th>fieldname</th>
<th>value</th>
<th>timesUsed</th>
<th>firstUsed</th>
<th>lastUsed</th>
<th>quid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>email</td>
<td><a href="mailto:testblackhat@devnull.com">testblackhat@devnull.com</a></td>
<td>1</td>
<td>13118230185430000</td>
<td>13118230185430000</td>
<td>CTwZ4359TYySue7E</td>
</tr>
<tr>
<td>2</td>
<td>enterAddressFull Name</td>
<td>Allan Smith</td>
<td>1</td>
<td>131182323235859000</td>
<td>131182323235859000</td>
<td>FXXpcowKuRqStorB-</td>
</tr>
<tr>
<td>3</td>
<td>enterAddressAddressLine1</td>
<td>42 my street</td>
<td>1</td>
<td>131182323235861000</td>
<td>131182323235861000</td>
<td>EFPAmGQESfryrz76n</td>
</tr>
<tr>
<td>4</td>
<td>enterAddressCity</td>
<td>San Francisco</td>
<td>1</td>
<td>131182323235861000</td>
<td>131182323235861000</td>
<td>/infoGyLjT4OmGoIQ</td>
</tr>
<tr>
<td>5</td>
<td>enterAddressStateOrRegion</td>
<td>CA</td>
<td>1</td>
<td>131182323235861000</td>
<td>131182323235861000</td>
<td>n8ckFxgx65SuaIt5Z</td>
</tr>
<tr>
<td>6</td>
<td>enterAddressPostalCode</td>
<td>94302</td>
<td>1</td>
<td>131182323235862000</td>
<td>131182323235862000</td>
<td>8ekzZSgQ6G+2mjc</td>
</tr>
<tr>
<td>7</td>
<td>enterAddressPhoneNumber</td>
<td>666-666-6666</td>
<td>1</td>
<td>131182323235862000</td>
<td>131182323235862000</td>
<td>8H1M4NVITGuBOcJ</td>
</tr>
<tr>
<td>8</td>
<td>enterAddressAddressAddressLine1</td>
<td>my street</td>
<td>1</td>
<td>131182323759915000</td>
<td>131182323759915000</td>
<td>hj8LVLsTFugo1eDV</td>
</tr>
<tr>
<td>9</td>
<td>searchBarHistory</td>
<td></td>
<td>1</td>
<td>131182328211160000</td>
<td>13118232821160000</td>
<td>po6Bcr72SydiGDN</td>
</tr>
</tbody>
</table>
Configuring a Linksys?
Again the key is recorded

<table>
<thead>
<tr>
<th>id</th>
<th>filename</th>
<th>value</th>
<th>timesUsed</th>
<th>firstUsed</th>
<th>lastUsed</th>
<th>quid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>wi0_wpa_psk</td>
<td>thisismywpakey</td>
<td>1</td>
<td>1311824553188000</td>
<td>1311824553188000</td>
<td>cw7adN3</td>
</tr>
</tbody>
</table>
Form history leak a lot of information

- Shipping address
- Wifi key
- Credit card information
- Email
- Search history
Preventing field recording

To tell the browser to not record a field use the tag

\texttt{autocomplete=\textasciitilde off}
• Passwords
• Location: registry
• Encryption: DPAPI + URL as salt
• History
• URLs: Index.dat
Decrypting Internet Explorer passwords
Decrypting Internet Explorer passwords

SHA1(URL) → Registry
Decrypting Internet Explorer passwords

Registry → SHA1(URL) → URL → URL List
Decrypting Internet Explorer passwords

Registry

\[\text{SHA1 (URL)} \rightarrow \text{URL (dpapi entropy)}\]

URL List
Decrypting Internet Explorer passwords

- SHA1(URL) → Registry
- SHA1(URL) → URL (dpapi entropy)
- URL → URL List
- URL → DPAPI Blob
- DPAPI Blob → Registry
Decrypting Internet Explorer passwords

1. SHA1(URL) → URL (dpapi entropy)
2. URL List
3. DPAPI Blob
4. Site password
5. Registry

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin
Beyond files recovery: OWADE cloud based forensic
http://owade.org
Maximizing our recovery

- Build a list of URL from others browsers and files
- Use a list of known login URLs
• Passwords
  • Location: Login Data (sqlite)
  • Encryption: DPAPI

History
  • URLs: History (sqlite)
  • Forms fields: Web Data (sqlite)
• **Passwords**

• Location: *keychain.plist* (Property list format)

• Encryption: **DPAPI** + fixed string as entropy

• **History**

• **URLs:** *History.plist*

• **Forms fields:** *Form Value.plist*
Browsers takeaway

• Internet Explorer is the most secure.
• If you don’t know the URL you can’t recover the credentials
• Firefox is the worst
  • Passwords encryption not tied to the Windows user password (bug open for a while)
  • Login are encrypted in signons.sqlite not in formhistory.sqlite
Private mode

• Most bugs are fixed
• Requires to be creative
  • SSL OCSP requests
  • File carving
• Potential techniques
  • Analyze the hibernate file

See: http://ly.tl/p16 for more information on private mode
### History

<table>
<thead>
<tr>
<th>URL</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>google.com</td>
<td>375</td>
</tr>
<tr>
<td>live.com</td>
<td>41</td>
</tr>
<tr>
<td>facebook.com</td>
<td>35</td>
</tr>
<tr>
<td>neutral.fr</td>
<td>31</td>
</tr>
<tr>
<td>skype.com</td>
<td>30</td>
</tr>
<tr>
<td>microsoft.com</td>
<td>28</td>
</tr>
<tr>
<td>aol.com</td>
<td>26</td>
</tr>
<tr>
<td>youtube.com</td>
<td>25</td>
</tr>
<tr>
<td>asho.fr</td>
<td>21</td>
</tr>
<tr>
<td>twitter.com</td>
<td>20</td>
</tr>
<tr>
<td>doubleclick.net</td>
<td>16</td>
</tr>
<tr>
<td>gmodules.com</td>
<td>16</td>
</tr>
<tr>
<td>msn.com</td>
<td>15</td>
</tr>
<tr>
<td>clubic.com</td>
<td>11</td>
</tr>
<tr>
<td>rut.com</td>
<td>10</td>
</tr>
<tr>
<td>iso9enhanced.com</td>
<td>10</td>
</tr>
<tr>
<td>apple.com</td>
<td>9</td>
</tr>
<tr>
<td>hotmail.com</td>
<td>9</td>
</tr>
<tr>
<td>accela.com</td>
<td>8</td>
</tr>
<tr>
<td>bing.com</td>
<td>8</td>
</tr>
<tr>
<td>fcdrn.net</td>
<td>8</td>
</tr>
<tr>
<td>steampowered.com</td>
<td>8</td>
</tr>
<tr>
<td>aim.com</td>
<td>8</td>
</tr>
<tr>
<td>wikis.com</td>
<td>8</td>
</tr>
<tr>
<td>atdmt.com</td>
<td>7</td>
</tr>
<tr>
<td>sourceforge.net</td>
<td>7</td>
</tr>
<tr>
<td>cnet.com</td>
<td>7</td>
</tr>
<tr>
<td>mydigitallife.info</td>
<td>6</td>
</tr>
<tr>
<td>busesdrivers.com</td>
<td>5</td>
</tr>
</tbody>
</table>
Instant messaging
• Encryption custom
• Difficulty extreme
• Location registry + config.xml
Decrypting Skype passwords
Decrypting Skype passwords

Registry → DPAPI Blob → pre-key
Decrypting Skype passwords

Registry → DPAPI Blob → pre-key → AES key: SHA1(pre-key)
Decrypting Skype passwords

- Registry
- DPAPI Blob
- pre-key
- AES key: SHA1(pre-key)
- encrypted credential
- config.xml
Decrypting Skype passwords

1. DPAPI Blob
2. Registry
3. AES key: SHA1 (pre-key)
4. MD5(login\nskyper\npassword)
5. Encrypted credential
6. Login

Pass cracking

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin

http://owade.org
Decrypting Skype passwords

There is a John the ripper patch for that

AES key: SHA1 (pre-key)

encrypted credential

config.xml

MD5(login\nskype\npassword)

pass cracking

Login

DPAPI Blob

pre-key

Registry
Google Talk

- Encryption
  DPAPI + custom (salt)
- Difficulty
  Hard
- Location
  registry
Salt derivation algorithm overview
Salt derivation algorithm overview

String: 0xBA0DA71D
Salt derivation algorithm overview

String: 0xBA0DA71D → Windows account name

Registry
Salt derivation algorithm overview

String: 0xBA0DA71D  Windows account name

Registry

http://owade.org
Salt derivation algorithm overview

String: 0xBA0DA71D → Windows account name

Registry

Windows account name → computer Netbios name

Registry
Salt derivation algorithm overview

String: 0xBA0DA71D

Windows account name

Registry

computer Netbios name

Registry
Salt derivation algorithm overview

String: 0xBA0DA71D ➔ Windows account name

Windows account name ➔ computer Netbios name

computer Netbios name ➔ DPAPI Blob

DPAPI Blob ➔ Registry

Registry ➔ Registry

Registry ➔ Registry

Registry ➔ Registry

Registry ➔ Registry

Registry ➔ Registry
Salt derivation algorithm overview

- String: 0xBA0DA71D
- Windows account name
- computer Netbios name
- DPAPI Blob
- Key

Registry
Microsoft Messenger

- Encryption
  DPAPI or Credstore

- Difficulty
  Medium

- Location
  version dependent
### Windows Messenger by version

<table>
<thead>
<tr>
<th>Version</th>
<th>Storage</th>
<th>Encryption</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Registry</td>
<td>Base64 encoded</td>
</tr>
<tr>
<td>6</td>
<td>Credstore</td>
<td>Credstore</td>
</tr>
<tr>
<td>7</td>
<td>Registry x2</td>
<td>DPAPI x 2</td>
</tr>
<tr>
<td>Live</td>
<td>Credstore</td>
<td>Credstore</td>
</tr>
</tbody>
</table>
• Encryption
  DES
  key: substr(login. "dummykey", 8)

• Difficulty
  easy

• Location
  config.xml
- Encryption
  XOR
  key: 9
- Difficulty
  trivial
- Location
  user.config
• Encryption
  **Base 64 + XOR**
  key: fixed string

• Difficulty
  **trivial**

• Location
  user.config
Beyond files recovery: OWADE cloud based forensic

E. Bursztein, I. Fontarensky, J.M. Picod, M. Martin

http://owade.org

- **Encryption**
  - *Clear aka encryt-what?*
- **Difficulty**
  - *none*
- **Location**
  - *account.xml*
Pidgin

- Encryption
  Clear aka encryt-what?
- Difficulty
  none
- Location
  account.xml
• Encryption
  Custom

• Difficulty
difficult (offline)

• Location
registry
Paltalk encryption algorithm
Paltalk encryption algorithm

VolumeSerial Number

01234567
Paltalk encryption algorithm

VolumeSerial Number
01234567

Paltalk account name
myusername

Registry
Paltalk encryption algorithm

VolumeSerial Number
01234567

Paltalk account name
myusername

m0y | u2s3e4r5|n6a7me x 3

Registry
Paltalk encryption algorithm

VolumeSerial Number
01234567

Paltalk account name
myusername

m0y l u2s3e4r5n6a7me x 3

encrypted password
yyy z yyy z yyy z yyy z

Registry

Registry
Paltalk encryption algorithm

VolumeSerial Number: 01234567

Paltalk account name: myusername

Encrypted password:

\[
ci: \text{yyyy}_z - \text{asciiCode}(S-\text{BOX}_{n-i})
\]
Paltalk encryption algorithm

VolumeSerial Number → 01234567

Paltalk account name → myusername

01234567 → myusername

m0y l u2 s3 e4 r5 n6 a7 me x 3 → encrypted password

encrypted password → yyy z yyy z yyy z yyy z

ci: yyy z i - asciiCode(S-BOXn-i)

Registry

Registry

Key
• If your Skype password is strong we can’t recover it
• Gtalk and Paltalk are the only ones to use computer information
• 3rd party software are the least secure
All the credentials recovered by OWADE

http://localhost:8080/owade/result_passwords_1

Chrome
Login: owade
Password: rootroot
Domain: ashe.fr

Chrome
Login: project.owade
Password: rootroot
Domain: google.com

Safari
Login: owade
Password: rootroot
Domain: ashe.fr

Trillian
Login: project.owade
Password: rootroot

GTalk
Login: project.owade@gmail.com
Password: rootroot

Most used
Passwords
rootroot

Usernames
owade
project.owade
Cloud based forensic
Cloud modules

- Leverage the credentials and history extracted to get cloud-data
- Might be legal (or not)
- Only LinkedIn currently (more modules almost ready)
OWADE status

- Alpha stage
  - Tested on Ubuntu against XP windows
- Roadmap
  - Stabilizing the code
  - Modularize the code so you write your own modules
  - More cloud probes: Facebook, Flickr, Emails...
  - Windows Vista and 7 integration
Conclusion

- People moving to the cloud means more data that are harder to get
- Forensics needs to evolve to cope with this
- OWADE is the first tool dedicated to cloud forensic
  - Decrypt the 4 major browsers data
  - Decrypt Instant messaging credentials
  - Open-source
Thank you!
Download OWADE
http://owade.org

Follow-us on Twitter
@elie, @projectowade

Donate to OWADE to support it!